(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 987 868 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
22.03.2000 Bulletin 2000/12

(51) Int Cl.[7]: **H04L 29/06**, H04Q 7/32

(21) Application number: **99307294.1**

(22) Date of filing: **14.09.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **14.09.1998 US 153322**

(71) Applicant: **Phone.Com, Inc.**
Redwood City, CA 94063 (US)

(72) Inventors:
- **Schwartz, Bruce V.**
  San Mateo, CA 94402 (US)
- **Greer, Russell S.**
  Los Gatos, CA 95030 (US)
- **Boyle, Stephen S.**
  Fremont, CA 94539 (US)
- **Fox, Mark A.**
  San Mateo, CA 94403 (US)
- **Rossmann, Alain S.**
  Palo Alto, CA 94303 (US)
- **Lentezner, Mark G.**
  Mountain View, CA 94041 (US)
- **Laursen, Andrew L.**
  San Mateo, CA 94402 (US)
- **Sandman, Brad E.**
  Sunnyvale, CA 94086 (US)

(74) Representative: **Ablett, Graham Keith et al**
Ablett & Stebbing,
Caparo House,
101-103 Baker Street
London W1M 1FD (GB)

(54) Method and architecture for interactive two-way communication devices to interact with a network

(57) The present invention relates to navigation of the Internet by a two-way interactive communication mobile device capable of wireless communication, via a link server (300), with service providers or network servers on the Internet. After the mobile device has established a communication session with the link server over a wireless network (308), a control engine 320 in the link server is initiated and uses the computing resources of the link server device so as to be responsible for tasks that require considerable computing power and memory. such as processing of URL requests, interpretation of markup language files, management of data cache and variable states. Working with a message processor (315) in the server device, the control engine communicates with an interface engine in the mobile device using a compact data format that is efficiently transportable in the wireless data network. The interface engine typically performs tasks that do not require considerable computing power and memory, such as receiving input data from users, and the rendering of the compact data format received from the link server device, to cause the mobile device to display contents in the markup language files on a display screen.

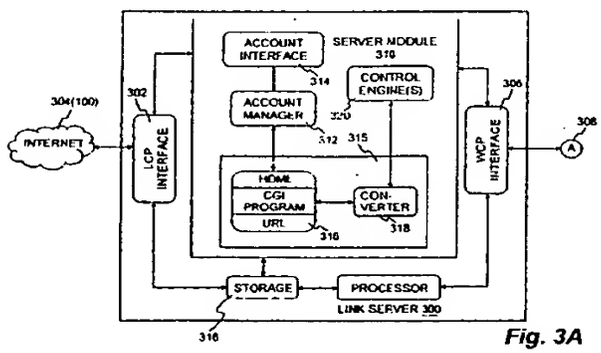

Fig. 3A

EP 0 987 868 A2

Description

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application is a continuation-in-part of pending U.S. Patent Application No. 08/570,210 entitled "METHOD AND ARCHITECTURE FOR AN INTERACTIVE TWO-WAY DATA COMMUNICATION NETWORK" by Alain Rossmann, one of the co-inventors hereof.

AUTHORIZATION WITH RESPECT TO COPYRIGHTS

[0002] A portion of the present disclosure contains material subject to copyright protection. Such material includes, but is not limited to, an Appendix entitled "Imp Specification protocols between Femto Engine and Terminal". The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF THE INVENTION

Field of the Invention

[0003] This invention relates generally to data communications, and in particular to interactive two-way communication mobile devices that permit a user to interact with a network server providing hypermedia information through a data network. Such a data network can include, for example, the Internet and a wireless network. The mobile devices may include cellular telephones, a two-way pagers, or a palm-sized computing devices and typically have limited computing resources.

Description of the Related Art

[0004] The Internet is a rapidly growing communication network of interconnected computers and computer networks around the world. Together, these connected computers form a vast repository of multimedia information that is readily accessible by the connected computers from anywhere at any time. To navigate a portion of the Internet organized as the "World Wide Web", the connected computers, e.g., workstations and desktop computers, typically utilize a user interface called a "browser". A browser is a client application program that generally requests multimedia information residing on the Internet using, typically, the Hypertext Transfer Protocol (HTTP). A computer which operates a browser using HTTP is generally a relatively powerful computer with sufficient computing resources, such as processing power, memory, a display capability and a user interface.

[0005] To provide mobility and portability of access to the Internet, interactive two-way communication mobile devices capable of communicating, via wireless data networks, with the Internet have been introduced. The interactive two-way communication mobile devices (e.g., two-way pagers, cellular phones, palm-sized computing devices and personal digital assistants (PDAs)) are among the fastest emerging communication devices. These devices enable users to receive, collect, analyse, review and disseminate information as the users travel or move about. Unlike computers coupled to the Internet, the mobile devices are characterized by severe limitations in computing resources. For example, a cellular phone has less than one percent of the processing power of a typical desktop personal computer, generally less than 128 kilobytes of memory, an LCD display which is perhaps four lines high by twelve or twenty characters, and limited or non-existent graphics capabilities. Further, a cellular phone inputs data using a keypad that has far fewer keys than a typical personal computer (PC) keyboard. With these constraints, a mobile device cannot efficiently operate the browser used by desktop computers to navigate the Internet.

[0006] To make available to mobile devices computing resources comparable to a desktop computer is too costly. There is, therefore, a great need for a solution that enables mobile devices to freely access information on the Internet without providing these computing resources in the mobile device.

[0007] Additionally, mobile devices are typically serviced through one or more wireless service carriers. The wireless service carriers often provide additional services by upgrading client application programs in the mobile devices. In conventional computers, an upgrade can be accomplished by downloading a new version of an application program from a service provider. In mobile devices, downloading a new version of an application program can be a prohibitively inefficient task, limited by the performances of the computing resources and the wireless network. Hence, there is further need for an ability to manage client application programs operated by the mobile devices.

SUMMARY OF THE INVENTION

[0008] The present invention addresses the above described problems and is particularly applicable to enabling navigation of Internet web pages by two-way interactive communication mobile devices (e.g., mobile computing devices, cellular phones, palm-sized computer devices, personal digital assistant devices and Internet-capable appliance remote controllers) which are capable of wireless communication via a link server with service providers or network servers on the Internet. Despite the common deficiencies of mobile devices (i.e., a primitive processor, little memory and limited graphics capability) which make it economically and technically impractical for the mobile devices to operate a local browser functioning as if it was in a desktop computer, the present invention allows the mobile devices to inter-

act effectively with the Internet and can be used with a wide variety of wireless communication networks (e.g., cellular digital packet data (CDPD) network, Global System for Mobile Communications (GSM) network, Code Division Multiple Access (CDMA) network and Time Division Multiple Access (TDMA) network).

[0009] According to one aspect of the present invention, a mobile device includes an interface engine that, via a client module, communicates and operates with a control engine in a link server device over a wireless network. The control engine, which utilizes the computing resources of the link server device, is responsible for tasks that require considerable computing power and memory, such as processing of URL requests, interpretation of markup language files, management of data cache and variable states. Further, working with a message processor in the server device, the control engine communicates with an interface engine using a compact data format that is efficiently transportable in the wireless data network. The interface engine typically performs tasks that do not require considerable computing power and memory, such as receiving input data from users, and the rendering of the compact data format received from the link server device, to cause the mobile device to display contents in the markup language files on a display screen.

[0010] According to another aspect of the present invention, incoming messages to the mobile device, including notification and requests, and which typically has one or more universal resource identifiers or locators, are processed in the link server device to generate compact messages. The link server device replaces universal resource locators in the incoming message with address identifiers, and manages an address table mapping each universal resource locator with an address identifier. Thus processed, the resulting compact messages demand less bandwidth of the wireless network, thus reducing high latency and requiring less air time.

[0011] According to still another aspect of the present invention, local service requests in the mobile device are processed simultaneously in the interface engine and the control engine. In the prior art, all local service requests are processed locally at the terminal where the local services are requested. The computing devices of the prior art, such as personal computers and workstations, can process local requests because of their computing power, memory and display capabilities. The mobile devices in the present invention, however, taking advantage of a cooperation between the interface engine and the control engine over the wireless network, services the requests with the limited computing resources of the mobile devices and without significantly compromising overall performance.

[0012] Thus, unlike the closed and proprietary prior art mobile devices (e.g., mobile phones and two-way pagers), the present invention allows thinly designed mobile devices to become open application platforms. Such an open application platform allows software developers to develop value-added applications and services to these thinly designed mobile devices. Consequently, many more new uses can be developed for two-way communication mobile devices and two-way communication networks, including wireless networks, without physical modification or addition to the two-way communication mobile device.

[0013] The present invention can be implemented in numerous ways. For example, according to one embodiment of this invention, a method of the present invention allows an interactive two-way communication mobile device with a display screen to interact with a network server. In this method, a control engine in a link server is initiated when the mobile device establishes a communication session with the link server. (Such a link server couples the network server of a landnet, which uses a first communication protocol, to a wireless network which uses a second communication protocol.) The link server includes: (a) an account manager managing a user account associated with the mobile device; and (b) a message processor receiving a message from the network server over the landnet. Upon initiation, the control engine communicates with an interface engine of the mobile device corresponding to the user account, and converts the message, using the message processor, to a compact data file that can be efficiently transportable in the wireless network.

[0014] According to another embodiment of this invention, in a method of the present invention, the link server sends over the wireless network a compact data file it generates, and the interface engine renders the compact data file to cause the display screen to display according to the content of the compact data file.

[0015] According to still another embodiment of this invention, a system of the present invention includes: (a) a memory for storing code for a server module; (b) a data storage device for maintaining a user account for the mobile device; and (c) a processor coupled to the memory and the data storage device. The processor executes the code in the memory to cause the server module to: (a) execute a control engine associated with an interface engine of a mobile device; (b) receive a network message from the network server over the landnet, using a first communication protocol; (c) buffer the network message in a cache memory; (d) generate a compact message from the network message; and (e) send the compact message to the mobile device over the wireless network, using a second communication protocol.

[0016] According to still another embodiment of this invention, a system of the present invention includes: (a) a display screen; (b) an input means; (c) a memory for storing code for a client module; and (d) a processor coupled to the memory and controlling the display screen and the input means. The processor executes code in the memory to cause the client module to: (a) execute an interface engine when activating a prede-

fined key; (b) maintain the interface engine to work with a control engine operating in the link server device in concert; (c) receive a compact message from the link server device over a wireless network, wherein the compact message is generated by a message processor in the link server device according to a network message received from the network server over a landnet; and (d) render the compact message to cause the display screen to display contents in the network message.

[0017] Accordingly, one of the objects of this invention is to provide a generic solution to two-way communication mobile devices with limited computing resources that can enable them to effectively interact with a landnet such as the Internet.

[0018] Other objects, together with the foregoing are attained in the exercise of the invention in the following description and resulting in the embodiment illustrated in the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] The present invention will be readily understood by the following detailed description in conjunction with the accompanying drawings, wherein like reference numerals designate like structural elements, and in which:-

Figure 1 illustrates a schematic configuration in which the present invention may be practised;
Figure 2A depicts a block diagram of a typical GSM digital cellular phone that can be used in the data network of FIG. 1 to practice the present invention;
Figure 2B illustrates an internal functional block diagram of an exemplary digital cellular phone that may corresponds to the GSM digital cellular phone of Figure 2A;
Figures 3A and 3B illustrate functional block diagrams of a link server device and a mobile device according to an embodiment of the present invention;
Figure 4 depicts an account structure used in the description of the present invention;
Figures 5A and 5B illustrate respectively two exemplary screen displays on a display screen of a mobile device;
Figure 6 demonstrates an overview of a systematic configuration according to the present invention;
Figures 7A to 7G illustrate a series of screen displays to illustrate the navigation of the Internet through a mobile device according to the present invention;
Figure 8A demonstrates an address table per device to send an address identifier to an actual IP address over a wireless network;
Figure 8B demonstrates an address table managed by an account manager to maintain groups of address identifiers in a link server for all the mobile devices in communication with the link server; and
Figures 9A and 9G illustrate a process flowchart of the present invention according to one embodiment.

DETAILED DESCRIPTION OF THE INVENTION

[0020] Referring now to the drawings, in which like numerals refer to like parts throughout the several views. Figure 1 illustrates a schematic configuration in which the present invention may be practised. As shown in Figure 1, landnet 100 is a land-based network that may be the Internet, an Intranet or a data network of any private network. Coupled to landnet 100 are a personal computer (PC) 110 and a network server 104. Personal computer 110 may be a Pentium II-based desktop personal computer. Preferably, personal computer 110 runs a HyperText Markup Language (HTML) browser, such as Netscape Navigator from Netscape Communications Corporation (http://www.netscape.com) via landnet 100 using HyperText Transfer Protocol (HTTP) to access information stored in network server 104, which may be a workstation from SUN Microsystems Inc (http://www.sun.com/). The information stored in network server 104 may be hypermedia information including mobile data designed for mobile devices.

[0021] There are n mobile devices 106 serviced by airnet 102. Mobile devices 106 are interactive two-way communication devices (e.g., mobile computing devices, cellular phones, palm-sized computing devices with PDA (Personal Data Assistants) functionality and Internet-capable appliance remote controllers) which are capable of communicating wirelessly with antenna 108 via airnet 102. As shown, antenna 108 also represents a wireless carrier infrastructure that generally includes a base station and an operations and maintenance center. The base station controls radio or telecommunication links with mobile devices 106. The operations and maintenance center comprises a mobile switching center performing the switching of calls between the mobile devices and other fixed or mobile network users. Further the operations and maintenance center manages mobile account services, such as authentication, and oversees the proper operation and setup of the wireless network. Each of the hardware components and processes in carrier infrastructure 108 are known to those skilled in the art and thus are not described here to avoid unnecessarily obscuring aspects of the present invention.

[0022] Between landnet 100 and airnet 102 there is a link server device 114 functioning as a bridge between the two networks 100 and 102. Link server device 114, which is also referred to as proxy server or wireless data server or network gateway server, may be a workstation or a personal computer. Link server 114, which is loaded with many processes including compiled and linked versions implementing the present invention, couples airnet 102 to landnet 100 and performs many functions as described in more detail below. One of the functions that link server 114 performs is to facilitate the communica-

tion of mobile devices 106 with any of the devices coupled to landnet 100, including mapping or translating from one communication protocol in landnet 100 to another in airnet 102 or vice versa.

[0023] To facilitate the description of the present invention, Figure 2A depicts a typical GSM digital cellular phone 200 that can be used as one of the mobile devices 106 in the arrangement of Figure 1 to practice the present invention. Cellular phone 200 includes a small screen 202 and an extended phone keypad 204. Screen 202 is typically a LCD display capable of displaying perhaps four lines high by twelve or twenty characters with limited graphics capabilities. Extended phone keypad 204 includes, preferably, a regular phone keypad 206, a pair of generic keys 208 and 210 and positioning key 212. Generic keys 208 and 210 are used to activate soft keys displayed in screen 202 and positioningkey212 is to reposition an element indicator or a cursor to activate, for example, one of the hyperlinks displayed in screen 202. Generic keys 208 and 210 and positioning key 212 are not necessary in practising the present invention. These keys can be replaced by a set of designated keys in regular phone keypad 206 but provide preferred convenient means for a user to interact efficiently with the phone 200. Further, having a regular phone keypad is not a requirement to practice the present invention. Some of the mobile devices have no physical keys at all, such as those palm-size computing devices that use "soft keys" or icons for receiving user input data. In the following, unless otherwise specifically described, keys or buttons are generally referred to as either physical keys or soft keys.

[0024] Figure 2B illustrates a functional block diagram of digital cellular phone 200. Since each of the hardware components in digital cellular phone 200 is known to those skilled in the art, the hardware components are not described in detail. Besides keypad circuit 246 for keypad 204 and display drive 248 for display screen 202, the main components in digital cellular phone 200 also include a random access memory (RAM), a read-only memory (ROM) and a physical layer processor or microcontroller 128. According to one embodiment, compiled and linked processes of the present invention are stored in ROM 250 as a client module 252 and a support module 254. Upon activation of a predetermined key sequence utilizing keypad 204, physical layer processor 128 causes client module 252 to communicate with link server 114 of Figure 1 via a radio transceiver 256.

[0025] It is generally understood that a computing device equipped with an HTML browser using HTTP can access hypermedia information in a network server. However, HTTP requires considerable computing power and network bandwidth resources. For example, a request from a computing device to establish a communication session with a network server may require an exchange of a number of data packets. In addition to the resources required to implement HTTP, significant resources must be supported in the computing device to request, format, process and display information. This is not a significant disadvantage in many situations because the computing device, including personal computers and workstations coupled to a network operating HTTP, generally has sufficient computing power, memory and display capabilities.

[0026] Nevertheless, cellular phone 200 or mobile devices 106 of Figure 1 typically do not have the computing resources to implement HTTP to run an HTML browser. The computing power in cellular phone 200 or mobile devices 106 of

[0027] Figure 1 is typically less than one percent of a laptop personal computer's computing power, the memory capacity is generally less than 128 kilobytes and the graphics display capability is very limited. Cellular phone 200 or any of mobile devices 106 of Figure 1 is not a replacement of a desktop computing device or the combination of a wireless communication module and a personal computer. Further, making a mobile device, such as cellular phone 200, capable of navigating hypermedia information in a network server is a significant departure from prior art systems.

[0028] Referring now to Figures 3A and 3B, there are respectively shown functional block diagrams of a link server device and a mobile device according to an embodiment of the present invention. Link server device, or simply link server 300, that may represent link server 102 of Figure 1, is typically a server computer. Mobile device 350 may, for example, correspond to one of mobile devices 106 of Figure 1 or cellular phone 200 of Figure 2. To avoid obscuring any aspect of the present invention, well known methods, procedures, components and circuitry in link server 300 and mobile device 350 are not described in detail.

[0029] Link server 300 includes a landnet communication protocol (LCP) interface 302 that couples to landnet 304, and a wireless communication protocol (WCP) interface 306 that couples to a wireless network 308 via a carrier's infrastructure (not shown in the figure). LCP interface 302 implements a communication protocol operated in landnet 304. Generally, landnet 304 operates HTTP, so that LCP interface 302 is typically an HTTP interface. Similarly, wireless network 308 may operate a wireless communication protocol suitable for the characteristics of a wireless network. One of the available wireless communication protocols is Handheld Device Transport Protocol (HDTP) (formerly known as Secure Uplink Gateway Protocol (SUGP)), which runs on User Datagram Protocol (UDP). In this embodiment, WCP interface 306 is implemented with a UDP or HDTP interface. HDTP is developed by Unwired Planet (Phone.com) Inc. located at 800 Chesapeake Drive, Redwood City, CA 94063. The specifications of HDTP, entitled "HDTP Specification" is enclosed and incorporated herein by reference in its entirety.

[0030] To facilitate the description of the present invention, the wireless communication protocol in use is

HDTP. The present invention is, however, not limited by this exemplary communication protocol.

[0031] HDTP is a session-level protocol that resembles HTTP but runs on UDP and without incurring the overhead of HTTP/TCP and is highly optimized for use in thin devices, such as the mobile devices, that have significantly less computing power and memory than those of a desktop personal computer. Further, UDP does not require a connection to be established between a client device and a server before information can be exchanged, which eliminates the need of exchanging a large number of packets during a session creation. Exchanging a very small number of packets during a transaction is one of the desired features for a mobile device with limited computing power and memory to effectively interact with a landline device.

[0032] Link server 300 further comprises a server module 310 coupled between LCP interface 302 and WCP interface 306. Server module 310, which is typically loaded in a memory, performs traditional server processing as well as protocol conversion processing from one communication protocol to another communication protocol. In particular, the protocol conversion processing includes protocol conversion between HDTP/UDP and HTTP/TCP according to one embodiment.

[0033] In server module 310, account manager 312 manages through account interface 314 a number of user accounts for all the mobile devices serviced by link server 300. Each of the mobile devices, such as 350, is assigned a device identification (ID). Device ID can be a phone number of the device or an IP address or a combination of an IP address and a port number, for example: 204.163.165.132:01 905 where 204.163.165.132 is the IP address and 01905 is the port number. The device ID is further associated with a subscriber ID created and administrated by a carrier in link server 300 as part of the procedures to activate a subscriber account for mobile device 350. The subscriber ID may take the form of, for example, 861234567-10900_pn.mobile.att.net by AT&T Wireless Service, and is a unique identification to a mobile device. In other words, each of mobile devices 106 serviced by link server 114 in Figure 1 has a unique device ID that corresponds to a respective user account in link server 114. Additionally, account manager 312 is responsible for creating a user account for a mobile device that anonymously communicates with link server 114. In this case, account manager 312 ensures proper (limited) access of the anonymous mobile device to services provided by link server 114.

[0034] Figure 4 shows an exemplary structure 400 of the user accounts managed by account manager 312. It should be noted that the user accounts need not be physically located in link server 300. In fact, the user accounts can be remotely located in one of the computing devices coupled to the landnet 104. Through account interface 314 that has proper and secure access to the user accounts, account manager 312 can conduct the duties of account management, as discussed in further detail below. Device ID column 402 is filled with the device IDs of mobile devices that correspond to subscriber IDs in subscriber ID column 404. Credential information column 406 lists credential information needed to access each associated account. User info 408 may include the account configuration information, for example, device ID "6508171453" is a mobile phone that is pre- configured to work in a CDPD network and, probably, may be provided with an option to switch to a GSM network if necessary. Further entries in user info column 408 may include pointers or linkages 410 to other account-related information, such as system parameters, encryption schemes, call plan and customer service information that can be accessed by the mobile device.

[0035] Returning now to Figures 3A and 3B, a database of user accounts permits account manager 312 to authenticate and to verify the subscribed mobile devices and to control access to provided services by all mobile devices (subscribed or anonymous devices) via wireless data network 308. More importantly in the present invention, account manager 312 is responsible for managing the operations of control engines 320, which are respectively and independently associated with one mobile device. The detailed operations of control engines 320 are provided below.

[0036] The following description is focussed on mobile device 350 and its associated account. However, the present description is equally applicable to any mobile device in communication with link server 300.

[0037] In addition, server module 310 includes message processor 315, which includes a message digester 316 and a converter 318. Message processor 315 processes messages communicated between a network server and link server 300 and generates for each message a corresponding compact message to be communicated between link server 300 and mobile device 350. In particular, message digester 316 receives the messages from the network server and performs a sequence of message processing that include interpretation and management of the messages. Converter 318 converts the messages, according to the interpretation, to a data format that is compact enough to be optimally efficiently transportable over wireless network 308. The messages received from the network server are typically markup language files or data, requests, notifications and other commands that could cause mobile device 350 to respond as desired in the received messages. The markup language may include, for example, Handheld Device Markup Language (HDML), HyperText Markup Language (HTML), compact HTML, Wireless Markup Language (WML), Standard Generalized Markup Language (SGML) and Extensible Markup Language (XML).

[0038] For example, LCP interface 302 receives an HDML file from a financial network server that directs mobile device 350 to display a pre-designed screen, in

response to mobile device 350's request to the financial network server. The exemplary HDML file is listed as follows:

```
< HDML VERSION = 2.0 >
    <DISPLAY NAME >
        < ACTION TYPE = ACCEPT TASK = GO DEST = #card2>
            Dow has hit 20,000 today !
            Nasdaq has popped 20%.
            Detailed Financial Headlines
    </DISPLAY >
</HDML>
```

[0039] The screen display corresponding to this HDML file is shown in Figure 5A. If a user selects the "OK" soft key, a list of the detailed financial news packaged in one or more HDML files would be fetched (pulled) from the financial network server and displayed, as shown in Figure 5B. As used herein, a display screen or screen is the physical display apparatus in a device, such as a 4 by 20 character LCD screen in a mobile device or 2.5 inch by 3.5 inch touch LCD screen in a palm-sized computer. A screen display is the image presented on the display screen. As shown in Figure 5B, display screen 500 shows a list of choices with element indicator 510 pointing to the first choice. Pointer 512 indicates that screen display 508 has more items to be displayed but limited by the size of display screen 500.

[0040] As described above, mobile device 350 typically does not have the necessary computing power and memory to operate a browser in response to the HDML files. Therefore, an HDML file received is first analysed by message digester 316 and then converted through converter 318 into a set of screen commands that cause a mobile device, upon receiving the screen commands, to display the contents in the HDML file in response to the screen commands. Typically, the screen commands are expressed in a form of screen description data (SDD) that is rendered in an interface engine in mobile device 350. The following is an example of an SDD stream:

```
c353 c836 e003 446f 7754 0368 6173 5803 6869 74e0 0632 302c
3030 3057 0574 6f64 6179 e001 2152 0844 6574 6169 6c65 64e0
0946 696e 616e 6369 616c e009 4865 6164 6c69 6e65 73ff
```

which is considerably smaller than the corresponding HDML file. The "ASCII-like" representation of the above illustrated SDD file is:

```
type = screen seq-num = 54
<WRAP> "Dow" offset=4 "has" offset=8 "hit"
<WRAP> "20,000" offset = 7 "today"
<WRAP> "!" offset = 2 "Detailed"
<WRAP> "Financial"
< WRAP> "Headlines"
< end >
```

Transmission of a smaller data file is important in wireless data networks that are characterized with by low bandwidth and expensive airtime. According to one embodiment, the SDD file is a group of Imp data, the detailed specification of the Imp data is provided in the Appendix entitled "Imp Specification protocols between Femto Engine and Terminal", which is hereby incorporated by reference for all purposes in its entirety. There are a set of rules or grammars in the Imp data that an interface engine, upon rendering the Imp data, causes a screen to display the contents of the corresponding markup language file.

[0041] In other words, the actual data being exchanged between link server 300 and mobile device 350 is in SDD format, which is typically binary and can be communicated more compactly and efficiently over wireless network 308. Further SDD files can be directly rendered by an interface engine in mobile device 350 without further processing. Nevertheless, the above procedures are provided for illustrative purpose only and the present invention.is not limited by to the Imp data format. According to another embodiment, the message processor does not have a pair of separate message digester and converter, a markup language file in HDML, compact HTML or XML is received at the message processor and converted into a corresponding binary file that is much smaller in size and may be in Imp, cHDML, cHTML, or cXML, wherein "c" means stripped, compressed, compiled or converted version of the corresponding markup files.

[0042] To interact with mobile device 350, server module 310 further includes control engine 320. Control engine 320 works in conjunction with an interface engine in mobile device 350 and further with message processor 315 to interpret actions from mobile device 350 in the present embodiment. More detailed description of the interactions between the interface engine in mobile device 350 and control engine 320 in server module 310 are is given below.

[0043] Mobile device 350 includes a corresponding WCP interface 352 that couples to airnet 308 via a RF transceiver (not shown in the figure) to receive incoming and outgoing data signals. WCP interface 352 is implemented with a UDP interface, as is WCP interface 306, when wireless network 308 operates HDTP. When another wireless communication protocol is operated in wireless network 308, both WCP interface 352 and WCP interface 306 are readily implemented accordingly so that link server 300 and mobile device 350 can communicate with each other.

[0044] Device identifier (ID) storage 354 supplies a device ID to WCP interface 352. The device ID identifies a mobile device 350 and directly corresponds to the device ID in the user account in link server 300. In addition, mobile device 350 includes a client module 356 that performs many of the processing tasks performed by the mobile device 350. Such processing tasks include establishing a communication session with line server 300 via carrier network 308, requesting and receiving data from carrier network 308, displaying information on a

display screen 360, and receiving user input data. Specifically, client module 356 is coupled to WCP interface 352 to establish a communication session and to request and receive data. Additionally, Client module 356 operates, among other things, an interface engine 364 that typically receives the screen description data from link server 300 and causes display drive 260 to display on the display screen what is intended in the HDML file originally received from the network server.

[0045] As mentioned above, in prior art systems, terminal devices typically run a local browser such as the one available from Netscape or Microsoft to interact with the Internet. The present invention, however, uses an interface engine in a terminal device and a control engine in a proxy server. In other words, the present invention uses an interface engine demanding little computing resources in a wireless mobile device and a control engine utilizing sufficient computing resources resident in a server device to allow the mobile device to effectively interact with a network server. Further, working in conjunction with the control engine in the link server, the interface engine in the mobile device does not require a large amount of computing power or memory to cache, parse, process and display a markup language file.

[0046] To facilitate further description of the present invention, Figure 6 illustrates an overview of a systematic configuration according to the present invention and should be understood in conjunction with Figure 3. Figure 6 shows that a mobile device 602 communicates with a network server 604 via link server device 606. Network server 604, or sometimes called service server, may be any server on the Internet that provides accessible hypermedia information. Mobile device 602 and link server 606 may correspond respectively to mobile device 350 and link server 300 in Figure 3. Service server 604 having an IP address, for example, http://www.abcnews.com provides hypermedia information to network 608 so that any computing devices coupled to network 608 can access the information in service server 604.

[0047] According to one embodiment, the information in network server 604 is a World Wide Web page that may be authored in HDML and fetched over network 608 operating HTTP. From the perspective of mobile device 602 that ultimately receives the information, link server 606 receives the HDML files that are then processed by message processor 610 and converted to screen description data compatible with the device characteristics of mobile device 602. The device characteristics may include the type and size of display screen and other information passed over link server 606 when a communication session is established between mobile device 602 and link server 606. Generally, a request to establish the communication session can be initiated by either mobile device 602 or link server 606. During the process of exchanging authentication information, the data representing the device characteristics of mobile device 602 is received and maintained in link server 606 such that the screen description data is generated per in accordance with the device characteristics of mobile device 602. The detailed description of initiating the request and the processing of exchanging information so as to subsequently establish a secure and authenticated communication session is described in commonly assigned U.S. Patent Application No. 08/966,988 entitled "Method and System for Secure Lightweight Transactions in Wireless Data Networks" by Hanqing Liao et al, which is hereby incorporated by reference in its entirety.

[0048] With the established communication session, the screen description data are then forwarded to mobile device 602 over wireless network 614 operating a wireless communication protocol. Upon receiving and rendering the screen description data, interface engine 616 causes display screen 618 to display the information embedded in the screen description data.

[0049] Figures 7A through 7G show the processes of navigation requests by mobile device 602 of Figure 6 and fetching requested information from service server 604 and forwarding the information subsequently from link server 606 to mobile device 602.

[0050] Prior to describing Figures 7A through 7G, some of the features in HDML are discussed. Similar to HTML, HDML is a tag-based document markup language which includes a set of commands or statements specified in a card that defines how information is displayed on a small screen. Normally a number of cards are grouped into a deck that is the smallest unit of HDML information that can be exchanged between network server 604 and link server 606 over landnet 608. The HDML specification entitled "HDML 2.0 Language Reference" is enclosed and incorporated herein by reference in its entirety.

[0051] According to one embodiment of the HDML, there are four typical types of cards: a display card, a choice card, an entry card, and a no-display card. A display card gives information to be displayed to the user. The displayed content can include any one of, or any combination of text, image, and soft keys. A choice card displays a list of choices to the user. The choices are presented in a format specified on the choice card and are generally numbered sequentially. As explained above, the user selects a choice by depressing a corresponding key. An entry card is used to obtain input data from the user. An entry card displays one or more entry lines. The entry line, in this embodiment, can be used to receive either numeric or text data. A no-display card is a hidden card which is not displayed. The no-display card is normally used to execute an intermediate action and generally not known to a user. Regardless of its type, a card can contain text, soft keys and images.

[0052] In one aspect and from the perspective of a browser operating HDML, choice and entry cards prevent a user from moving to the next card until the requested information is received from the user. When the user reaches the last card in a deck and hits a corre-

sponding key, a request for a new deck is initiated. The deck requested is determined by either the deck that the user has completed, or by the choices made by the user. When the deck is completed, the choices and/or data entered by the user are typically transmitted along with the request to a network server for a new deck. When a deck containing multiple cards is received and stored in a cache memory, the browser fetches the first card in the deck, displays the information in the card, and allows the user to respond thereto. Depending on the card type, the user responds by entering text or choosing an option, and then pressing a predetermined key to transact the response.

[0053] Figures 7A through 7G should be understood in conjunction with Figure 6 and with reference to Figure 3. upon establishing a communication session between mobile device 602 and server device 604, an initial HDML deck HDML transmitted to link server 606 includes an introductory display card and a choice card. Figure 7A is an example of introductory screen display 702 that is ultimately drawn on a display screen 700 of mobile device 602 by interface engine 616. Figure 7A and the following figures are not interpreted directly from the HDML decks received, rather are interpreted from corresponding screen description data translated in link server 606 according to the HDML decks received therein. As described above, if working directly with the HDML files, the terminal (i.e., the mobile device) would require both considerable memory to cache the HDML files, history and activity states and sufficient computing power to run a browser to work with the cached HDML files. One aspect which differentiates the present invention fundamentally from prior art systems is that the control engine in the link server is responsible for tasks that require extensive computing resources while the interface engine in the terminal is only responsible for rendering the screen description data to cause the display screen to display contents and receiving inputs from a user. More specifically, the typical functions that the control engine in the link server perform include:

1. processing requests from the mobile device;
2. generating a URL request to a network server;
3. interpreting markup language files;
4. generating screen description data;
5. management of data cache;
6. management of history;
7. management of variable states in a markup language file;
8. maintaining push data, (including alerts), electronic mails.

[0054] According to one embodiment, display screen 700 displays a graphical image. In another embodiment, display screen 700 displays only text. Screen display 702, and other screen displays described more completely below, include a horizontal arrow 704, i.e., a multi-screen indicator translated from a multi-card deck indicator, to communicate to the user that screen display 702 includes another screen display. To view the HDML file, a multi-card deck indicator indicates that the current deck includes another card. The inclusion of screen indicators, such as horizontal arrow 704, to communicate with the user is optional. The functionality of this invention is independent of such screen indicators.

[0055] Referenced by 706 is a soft key generally associated with one of the generic buttons of the keypad of the mobile device 602. A soft key can be used to map a generic button into a specified button or activated by a touch pen or a finger. In this instance, pressing the generic button or touching the key directly is equivalent to pressing an "OK" button when the soft key OK is displayed. In many palm-sized computing devices, the number of the keys is generally kept toa minimum so as to provide a larger display screen. The larger display screen can accommodate more soft keys, which can be directly activated using a touch pen. Soft keys thus provide an efficient means to interact with display screen 700.

[0056] When the user depresses a predetermined key (i.e. one of the generic buttons in this case), thus selecting a soft key, a client module in the mobile device 602 interprets the action and sends a request to link server 606. Upon receiving the request, control engine 609 in link server 606 interprets the request which is, in this instance, a request to display the next screen display. Control engine 609 calls converter 612 to retrieve the next card from the received HDML deck, preferably, cached in a memory in the link server and converts the card in HDML to a SDD file that is subsequently delivered to mobile device 602. Upon receiving the SDD file, interface engine 616 draws a new screen display as shown in Figure 7B, which would be otherwise displayed on a desktop computer running a local browser working directly with the HDML file.

[0057] Screen display 708 in Figure 7B shows a list of choices (the original HDML card is a choice card). Besides the list of choices that can be accessed by the user in Figure 7B is a downward arrow, which indicates that the screen display includes additional items that are not shown in display screen 700. The screen display can be larger than the number of lines available in the display screen 700 and so the user must scroll the screen display to view the complete screen. Thus, to view the additional items, the user presses the downward arrow key corresponding to the downward arrow indicator 712 on the display screen 700.In this embodiment, when the downward arrow key is pressed, each line of the display is rolled up one line. The resulting display has an icon with an upward arrow (not shown) if the menu requires only two screen displays. If the menu requires more than two screen displays, the second screen display of the menu would have two icons, one with the upward arrow and another with the downward arrow. To scroll between the various lines in the second menu, the user uses the downward arrow key, and the upward arrow key. If the

user displays the last line of a card, e.g., the last line in the second menu, and presses the downward arrow key nothing would happen because the downward arrow icon, another soft key, will not be present. In this screen display, the user must make a choice before a next display screen is available.

[0058] In this embodiment, each of the menu items is available on service server 604 or distributed on several server computers coupled to network 608. As explained more completely below, each of the menu items in the original HTML file is associated with a numeral that corresponds to a resource locator in the card containing the menu items. The resource locator includes an address of a particular object associated with one of the menu items. In general, a resource locator includes a universal resource identifier (URI) or universal resource locator (URL) and may include appended data. The address can be referenced to another card in the deck cached in link server 606 or to a remote object on service server 604.

[0059] As shown in Figure 7B, the first item in the menu 708 is initially indicated by an arrow 710 as a prechosen item. If the user decides to proceed with the prechosen item, the soft key "OK" may be pressed. Alternatively, a numbered key "1", i.e. one of the 10 numbered keys, can be pressed to cause the client module in mobile device 602 to send a new request to link server 606 for a next screen display. This new request, however, is not a simple request as the one from Figure 7A. This request may include a resource locator to another card in the deck cached in link server 606 or a remote object in service server 604, depending on whether the original received HDML includes the information requested by the new request from mobile device 602.

[0060] This new request corresponds to a hyperlink in the card that has been converted to the SDD file currently being displayed in Figure 7B. The hyperlink may include a URL as follows:

www.xyzinfo.com/ABCcorp/sales

where www.xyzinfo.com can be the URL of service server 604 and /sales may be a hyperlink in an object identified by /Softcorp in service server 604. More specifically, the card in the original HDML file may be expressed as follows:

```
< HDML version = "2.0">
    < CHOICE >
        <CE TASK = GO DEST=www.abc.com/sales.hdml> ABC Corp. Sales
        <CE TASK = GO DEST=www.xyzinfo.com> XYZ Information
        < CE TASK = GO DEST =www.financialinfo.com> Financial Info
        < CE TASK = GO DEST = www.personalweb.com > Personal Web Site
        ...        </CHOICE>
</HDML>
```

In the present example, each of the items in the menu displayed in Figure 7B corresponds to a URL in the following, identifying a network server in the Internet:

www.abc.com/sales.hdml
www.xyzinfo.com
www.financialinfo.com
www.personalweb.com

When one item is subsequently chosen, the control engine will generate a URL request to the identified network server to retrieve the desired information.

[0061] Although converter 612 in link server 606 converts the above code to a SDD file, a much more compact format for transmitting over wireless network 614. A long address, like http://www.xyzinfo.com/LocalNews/Towns, typically cannot be compressed further. It is neither efficient nor wise to use the wireless network to communicate a number of long addresses in a file and return a URL request containing one or more of the addresses. Hence the present invention uses one or more address identifiers that are communicated over the wireless network. Each of the address identifiers identifies the full address. An address table is maintained in link server 606 that maps the address identifiers to the actual (full) addresses. The address identifying or address mapping methods described here are significantly different from prior art systems which send terminal device addresses to all hyperlinks in a markup language document along with the document to a terminal device.

[0062] Figures 8A and 8B show, respectively, two implementations of address mapping and should be understood in conjunction with Figure 3 and Figure 6. Address mapping table 800 is identified by a user ID 802. Address mapping table 800 includes address identifier column 804 and addresses into address buffer806. Address mapping table 800 in Figure 8A is typically established when a communication session between a mobile device and a link server is created. Each mobile device is allocated one address mapping table, which can be managed by the account manager in the link server. In other words, user ID 802 (e.g., a device ID or a subscriber ID) uniquely associates a mobile device to address mapping table 800. During the communication session, only the entries in address identifier column 804 are actually sent. For example, rather than sending over the entire resource locator http://www.xyzinfo.com/LocalNews/Towns, address identifier "1234" is embedded in a SDD file that is delivered to the mobile device. Generally mapping table 800 in Figure 8A vanishes when the session expires or terminated.

[0063] According to another embodiment, the account manager manages an address mapping table 800 shown in Figure 8B, in which user ID column 802 includes identifications (e.g. device ID or subscriber ID) of all active mobile devices communicating with the link server. Address identifiers 804 includes all address identifiers corresponding to the actual addresses in address buffer 806. Thus, whenever a mobile device refers by using an address identifier to a resource locator, the

actual address is retrieved from address buffer 806 using the address identifier and used in a URL request generated by the control engine to the identified network server.

[0064] According to another embodiment in which the SDD is a group of Imp data, the actual addresses are mapped to their relative positions in a final screen display. For example, the above four URLs are hyperlinks, according to the original HDML file, to be displayed one in each of successive lines. Thus their relative positions, linel, line2, line3 and line4, each corresponds to one of the URLs. The relationships between the relative positions and the actual URLs may be maintained in the address table discussed above or directly by the control engine. If a user eventually chooses one of the hyperlinks, a (client) request from the mobile device will comprise include the chosen position. The request may be expressed as follows:

client request = (SeqID, link, topline);

where "SeqID" ensures that the client request is synchronized with the Imp data fetched for the mobile device, from which the client request is produced, "link" is one of the parameters that indicates which link (URL) is chosen and "topline" is the position in the screen display from the Imp data. For example: client request = {64, 2, 0}; Upon receiving the client request, the control engine processes the request and produces an updated request including the actual URL corresponding to the chosen position, for example {"http://www.xyzinfo.com"}, which causes a connection to the service server identified by the URL.

[0065] Returning to Figure 7B, the user may scroll the choice arrow 710 downward if the pre-chosen item is not a desired one. It should be noted that scrolling to a selected item is a feature that is specific to this example, and in general is not required to implement the invention. Other methods can be used to indicate the user's choice on display screen 700 such as a horizontal highlighting strip overshadowing the choice, if such an indication is desired. As described above, the user may simply key in one or more numerals to select an item that is of interest.

[0066] As described above, screen display 716 also includes the representations of two soft keys, an OK key 706, and a Back key 714. In this example, these soft keys are defined only for the card used to generate screen display 716. The "OK" key allows the user to proceed with the chosen item and the "Back" soft key allows the user to go back the previous screen display if so desired. In the present invention, the "Back" soft key may generate a request that is sent over to the link server from which the previous screen display is fetched again. Other keys can be implemented. For example a "Home" key, resulting in a request that returns the user to screen display 708 of Figure 7B. The "Home" key may be associated with a resource locator identifying the card representing screen display 708. Specifically, the link server manages a limited history stack of recent requests made by the mobile device in a memory. When a request is made, the control engine looks up the history stack to see if the request is an "old" one. For example, when the "Home" key is pressed, the request can be found in the history stack and the contents, either in the form of an HDML card or an SDD file, can be retrieved from memory and forwarded to the mobile device for display.

[0067] As shown in Figure 4C, the user moves arrow 710 downward to the second item. Display screen 716 shows four menu items numbered consecutively. As described above, the downward arrow indicates that there are more items in the next screen. Each of the items has an address identifier. For example, for the first four items, the respective address identifiers may be:

12ab
231a
abc3
1629

each address identifier correspond to an address stored in the address buffer of link server 606:

www.abc.com
www.xyzinfo.com
www.financialinfo.com
www.personalweb.com

When the second item (i.e., 231a) is chosen, www.xyzinfo.com is intended. After a predetermined button is pressed (e.g., the soft key OK or the numbered button "2") is pressed, a request including the address identifier for the selection is transmitted to link server device 606 by the client module in mobile device 602 over network 614. Alternatively with regard to the specific Imp data implementation, the request includes the selection in terms of the relative position in a display screen as described above. In response to the selection, control engine 609 processes this request and formulates a new or updated request containing the actual URL identified in the client request, which causes a connection to service server 604. Through the server module, the link sever receives another HDML deck file from service server 604. Upon receiving the new HDML deck, message processor 610 processes the deck for the desired card and sends a corresponding SDD file derived from the desired card to mobile device 602.

[0068] In Figure 7D, there is shown a new screen display 718. Typically, it is based on one of the cards in a new deck received in the link server as a result of the request from screen display 716. The deck is cached in the link server and the first choice card is converted to a SDD file that is rendered by the interface engine in the mobile device for display. If the user proceeds with any of the items, for example, "Local News", a request is made from the interface engine in the mobile device and received by the corresponding control engine in the link server. The control engine causes the message processor to retrieve a card identified by the request from the cache and convert the card to a SDD file and forwards the file to the mobile device for display.

[0069] Figure 7E shows a display screen 718 result-

ing from the "Local News" request. Display screen 718 asks the user for specific date information so that the news corresponding to the specified date can be provided. The original HDML card that corresponds to display screen 718 is an entry card that requires an input from the user. Hence the corresponding SDD file converted from the HDML entry card requires the input at cursor 720. Figure 7F shows that the input 722, i.e. date information, is typed in. Upon pressing soft key "OK" 706, the interface engine sends a request including the input data over to the control engine that performs variable substitutions. Variable substitutions, permitting sharing of data between cached cards, substitute the variable in the original HDML card with the actual information. As a result, an updated HDML card is locally and dynamically generated and then converted to a new SDD file that is returned to the interface engine for display. Figure 7G shows a screen display 724 with the substituted data information. In this example, the updated HDML card is another entry card, hence screen display 724 asks for further information in order to deliver accurate information to the user. If the user supplies the "town" information being requested and presses the "OK" soft key, a request is made and sent to the link server in which the supplied information is used to substitute a corresponding variable and an updated request with the date and town information is generated. Typically, the updated request is sent to the network server supplying the information, but the updated request may be filled locally in the link server if the original HDML deck is large enough to include the desired information. Further detailed description of the management and processing of the variables in a markup language file is provided in commonly assigned U.S. Patent Application No. 09/071,235 entitled " Method for Inline Variables Management in a Hypermedia Display Language" by Peter F. King et al, which is hereby incorporated by reference in its entirety.

[0070] Figures 9A to 9G together constitute a process flow diagram showing the process performed by a link server device and a mobile device according to one embodiment of the present invention and should be understood in conjunction with Figures 3A to 3B and Figure 6. At 904, link server device exchanges information with the mobile device to establish a communication session. The request to establish a communication session is initiated from the link server by sending a message to the targeted mobile device. The message includes a device identification of the mobile device. Upon receiving the message, the mobile device starts exchanging information with the link server. The exchanged information may establish the encryption keys and the encryption scheme to be used for the session. In addition, the mobile device delivers to the link sever a set of device characteristics information regarding the type and size of the display screen of the mobile device. At 906, the account manager in the link server associates the device information with the session just established. Typically the device information is cached in a memory along with other information about the mobile device. If the mobile device is an authorized device, there is a corresponding account that was established when the mobile device is activated. If the mobile device contacts the link server the first time, an account is established by the account manager. Therefore, the device characteristics information is always associated with the account of the mobile device.

[0071] At 908, the account manager assigns a control engine to work in conjunction with the interface engine in the mobile device. At 910, the account manager detects, through the server module, any message arrived. At 912, the source of the message is identified (i.e., whether the message is received from a network server or from the mobile device).

[0072] At 914, when the received message is from the network server, the control engine along with other modules in the link server-determines the message type. In this embodiment, there are broadly two message types that are processed distinctively from the prior art systems. Specifically, these message types are notifications and markup language (ML) files. The notification or alert message indicates the arrival of an electronic mail or fulfilment of certain requests (e.g., sale of a stock at a limit price). The notification or alert message includes a device identification identifying the mobile device, an alert type (instructing the mobile device to beep, vibrate or display a visual sign), an alert title (a text string describing the subject matter of the alert), a life-time specifying a time period during which the alert should be delivered) and a URL that a user can request when the user desires to respond to the alert. Alternatively, an alert can be expressed as follows:

Notificationalert = {023, "new mail", 4,www.wireless.com/mail_retrieval/87473} where "023" is a special code that can causes the mobile device to beep, the title "new mail" is then displayed on the screen of the mobile device, the value "4" specifies that the notification message be delivered within four hours or discarded, and the last entry in the notification is the URL to retrieve the new mail identified by "87473" from a mail server identified by www.wireless.com.

[0073] As indicated above, a notification or alert message is not always immediately deliverable; sometimes the mobile device is out of the service area or the mobile device is turned off. Consequently, the account manager of the link server maintains a notification list or an alert list for each mobile device. Upon receiving a new alert message, at 916, the account manager determines from an alert list if the newly arrived alert message corresponds to a URL already on the alert list. If there is an identical URL in the alert list, at 920, the corresponding entry in the alert list is updated with the newly arrived alert message. If no identical URL is found, at 922 the newly arrived alert message is inserted. The newly arrived alert messages are sequenced in the alert list for delivery to the target mobile device.

[0074] At 924, the alert message is modified by substituting the actual URL by using an address identifier retrieved from an address table. At 926, the modified alert message is sent to the mobile device over the wireless network. It should be pointed out that the above alert list update is not necessary if the newly arrived alert message is immediately delivered. Further it should be pointed out that the alert list may not be necessarily maintained in the link server device and, as will be explained below, may be maintained in the mobile device.

[0075] Returning to 914, the situation when the received message from the network is a markup language (ML) files is described. At 938, the message processor in the link server processes the ML files. The processes at 938 may include caching the ML files in proper memory, parsing the ML files to generate internal data structure needed to generate SDD files. In particular, at 940 and 942 all the URLs in the received ML files are substituted with corresponding address identifiers, with the actual URLs are stored in the address table maintained in the link server or the relative positions of the URLs are determined with regard to the Imp data implementation. At 944, the message processor converts the processed ML files to SDD files based on the mobile device's characteristics information, to allow proper display of the SDD files in the mobile device. To ensure that the control engine in the link server is in synchrony with the interface engine in the mobile device, at 946, the SDD files are respectively sequenced, preferably numbered consecutively and, at 948, delivered to the mobile device over the wireless network.

[0076] Returning to 912, the situation when the message is from a mobile device is described. Typically, such a message includes one or more (client) URL requests. At 960, the control engine processes the message after the account manager verifies that such requests are permissible at 958. Depending on the services subscribed, each mobile device serviced by the link server may have the different privileges from other mobile devices to the services offered by the link server. If a request is granted at 958, the link server processes the request. Collectively, a (client) request may be expressed as follows:

client request = {SeqID, Event, Choice, Link, AlterID, Topline, Entry, URL} where "SeqID" ensures that the client request is synchronized with a SDD fetched to the mobile device, from which the client request is produced. "Event" indicates what kind of request this client request is, for example, Softkey meaning a soft key activation, AlertSelect meaning that an alert has been responded to fetch a message in reference to the alert, and "Accept", "GotoURL" and "DeleteSelect", to name a few. "Choice" indicates which choice in a screen display has been chosen. "link" is one of the parameters that indicates which link (URL) is chosen. "AlterID" comprises one or more those address identifiers. "topline" is the position in the screen display from the SDD and "Entry" typically holds inputs entered by a user and "URL", as the name suggests, holds an address enter by the user.

[0077] At 960, the request is examined. In some instances, at 962, variables in the requests are substituted to provide an updated request, as explained below.

[0078] According to one aspect of the invention, variables are used to hold user input data. Such user input data can be collected, for example, in response to a query provided to the user in a display screen. When the user input data (e.g., a number) is entered by the user, the user data received is provided on the next display screen to provide feed back to the user. Specifically, the link server receives an ML file in which are defined a number of variables. The variables in the ML file constitute information to be requested at a terminal device. The ML is converted to a corresponding SDD file to be displayed on the mobile device, and in response to the display screen, the user enters the required input data and a request is dispatched containing the input data to the link server, after a predefined key is pressed. In prior art systems, a terminal device running a browser performs the substitutions locally. In the present invention, the mobile device operates only an interface engine without the capability of performing the substitution. The substitutions are instead performed by the control engine in the link server when the request from the mobile device is received at 964. The link server responds to the request by sending the mobile device a new SDD file that includes the user data substituted for the variables at 966.

[0079] According to another aspect of the present invention, some values received from the mobile device for variables in the ML files are in the form of address identifiers that must be substituted with the actual URLs. Examples of a request including address identifiers include a new information request to a network server or a request to retrieve email. Upon receiving such a request, the actual URLs are retrieved by the account manager from an address table in the link server. At 968, the original requests from the mobile device are modified to produce updated requests with the actual URLs substituted and the updated requests are then sent to the identified network server(s) corresponding to the URLs at 970.

[0080] Figures 9E to 9G constitute a process flow diagram of the operations executed in the mobile device, which corresponds to the processes in the link server. At 953, the mobile device exchanges information with the link device to establish a communication session. The request to establish a communication session can also be initiated from the mobile device by sending a message to the link server. Besides a device identification of the mobile device, the message includes a URL of the link server. To establish the communication session, device characteristics information is provided to the link server. Such characteristics information may include the size and type of the display screen of the mobile device, for example. After the communication ses-

sion is established, the interface engine works at 957 with the control engine in the link server.

[0081] At 959, the client module in the mobile device receives a message. Typically the mobile device receives three kinds of messages: notifications, SDD files and local service requests. At 963, a notification message arrives. Note that a notification message received at the mobile device is different from the notification or alert message provided by a network server. The notification message received at the mobile device is a distilled version with no explicit URLs. Upon receiving the notification message, the client module accesses an alert list in the mobile device to determine if there is an identical notification pending there at 965. Sometimes a user of the mobile device may not necessarily or immediately respond to an alert, the mobile device hence maintains an alert list to keep record all the received notification or alerts. If a pending notification identical to the newly received notification message is found, the alert list gets updated with the newly arrived notification message at 967. Otherwise, i.e., no identical notification message is found in the alert list, the newly arrived notification message is added sequentially into the alert list at 969. Meanwhile, the user is notified in accordance with the alert type in the received notification message at 971. When the user decides to respond to the notification and presses a key or activates a soft key, a request corresponding to the notification message is sent to the link server at 973.

[0082] At 975, a message is received requesting an update to the local services in the mobile device. Local services may include functions for modifying wireless voice/date protocols, configuration or system parameters, bookmarks, addresses, subscriber provisioning information and other parameters that may enable or disable certain telephony and data-features of the mobile devices. Technically, the interface engine recognizes such a message by a special prefix indicating the "local service" request. According to one embodiment, a URL for a local service always begins with "device:". (e.g., device:addressbook).

[0083] Upon the arrival of a local service request, the local service in the mobile device is invoked. For example, a user may navigate to a page providing email service to the mobile device. After a key is pressed or a soft key is activated, a request is sent to the control engine of a link server, which in turn responds by a local service request which causes an address book to be displayed in the mobile device. After the user makes a selection from the address book, at 977, the mobile device sends another request specifying the selected address to the control engine which then sends the mobile device an SSD file. Upon receiving the SSD file, the display screen displays a page which allows the user to proceed with composition of a mail message.

[0084] At 981, the situation when the received message is an SDD file is described. Upon receiving the SDD file, the interface engine renders the SDD file and causes the display screen of the mobile device to display text or graphics according to the SDD file at 983. Within the display screen, the user may browse the screen display at 985 by pressing a navigation key to reposition a cursor to a subject of interest. For further information on the selected subject, the user may press a predefined key; hence a URL request is generated at 987. Also at 985, the user may be asked for input data to some context. Once the input data is entered, the user may press a predefined key, such as the "OK" key to generate a URL request at 987. The URL request is then sent to the link server for processing at 989.

[0085] The present invention is described above by way of example using specific embodiments. Numerous changes and modifications can be made within the scope of the invention claimed below.

Claims

1. A method for an interactive two-way communication mobile device having a display screen and communicating with a wireless network to interact with a network server, the method comprising:-

   initiating a control engine in a link server device of a landnet after the mobile device establishes a communication session with the link server device over the wireless network, wherein the link server device comprises an account manager configured to manage a user account of the mobile device; and a message processor configured to receive a message from the network server over the landnet;
   associating the control engine with an interface engine operating in the mobile device corresponding to the user account; and
   operating the message processor to convert the message to a compact data file that can be efficiently transported in the wireless network.

2. A method as recited in claim 1 further comprising forwarding device characteristics information of the display screen from the mobile device to the link server device over the wireless network.

3. A method as recited in claim 2 wherein the compact data file is screen description data that can be directly rendered by the interface engine in the mobile device.

4. A method as recited in any preceding claim wherein the message received from the network server is a notification comprising a device identifier identifying the mobile device and a universal resource identifier.

5. A method as recited in claim 4 wherein said con-

verting the message to a compact data file by the message processor comprises:-

looking up in a notification list managed by the account manager for an entry substantially equivalent to the received notification in the link server device;
updating the notification list with the received notification;
substituting the universal resource identifier with an address identifier;
storing the universal resource identifier and the corresponding address identifier in an address table managed by the account manager in the link server device; and
converting the notification with the universal resource identifier substituted by the address identifier to the compact data file.

6. A method for an interactive two-way communication mobile device of a wireless data network to interact with a network server, the mobile device having a display screen to interact with a network server, the method comprising:-

establishing a communication session between the mobile device and a link server device over the wireless network, the link server device being coupled to the network server through a landnet so that the mobile device interacts with the network server via the link server device;
associating an interface engine operating in the mobile device with a control engine operating in the link server device with an account established for the mobile device in the link server device;
receiving a compact data file generated in the link server device over the wireless network; and
rendering the compact data file by the interface engine to display contents of the compact data file on the display screen of the mobile device.

7. A method as recited in claim 6 wherein said establishing a communication session comprises forwarding device characteristics information of the display screen of the mobile device to the link server device over the wireless network.

8. A method as recited in claim 6 or 7 wherein the compact data file is an updated notification processed in the link server device from a notification received from the network server, the notification comprising an alert type and an universal resource identifier.

9. A system, coupling a wireless network to a landnet, for an interactive two-way communication mobile device having a display screen to interact with a network server, wherein the mobile device is coupled to the wireless network and the network server is coupled to the landnet, the system comprising:-

a memory for storing code for a server module;
a data storage device for maintaining a user account for the mobile device;
a processor coupled to the memory and the data storage device, the processor executing the code in the memory to cause the server module to:-
maintain executing a control engine associated with an interface engine executing in the mobile device;
receive a network message from the network server over the landnet operating a first communication protocol;
buffer the network message in a cache memory;
generate a compact message from the network message; and
send the compact message to the mobile device over the wireless network operating a second communication protocol.

10. A system as recited in claim 9 wherein the processor executing the code in the memory further causes the server module to establish an account manager to manage a user account of the mobile device; the account ensuring that the control engine operates with the interface engine in concert.

11. A system as recited in claim 9 or 10 wherein the network message is a notification comprising an alert type and a universal resource identifier.

12. A system for interactive two-way communications with a network server through a link server device, the system comprising:-

a display screen;
an input interface;
a memory for storing code for a client module;
a processor coupled to the memory and commanding the display screen and the input means, the processor executing the code in the memory to cause the client module to:-
execute an interface engine when activating a predefined key;
maintain the interface engine to work with a control engine operating in the link server device in concert;
receive a compact message from the link server device over a wireless network, wherein the compact message is generated by a message processor in the link server device with reference to a network message received from the network server over a landnet; and

render the compact message to cause the display screen to display contents in the network message.

13. A system as recited in claim 12 wherein the input interface is a phone keypad.

14. A system for interactive two-way communications with a network server of a landnet operating a landnet communication protocol, the system comprising:-

a plurality of interactive two-way communication mobile devices, each having:-

a display screen;
an input interface;
a memory for storing code for a client module; and
a microcontroller coupled to the memory, the microcontroller executing the code in the memory to cause the client module to maintain an interface engine therein;

a link server device coupling the landnet and a wireless network operating a wireless communication protocol; the link server device comprising:-

a memory for storing code for a server module;
a processor coupled to the memory, the processor executing the code in the memory to cause the server module to:-
create a manager managing a plurality of user accounts, each with respect to a device ID of one of the mobile devices;
maintain a plurality of control engines, each being associated with the interface engine in each of the mobile devices after a communication session is respectively established between the link server device and the each of the mobile devices over the wireless network;
receive a network message from the network server over the landnet; the network message comprising information of the device ID of the one of the mobile device;
generate a compact message from the network message according to display characteristics information of the one of the mobile devices; and
send the compact message to the one of the mobile devices over the wireless network.

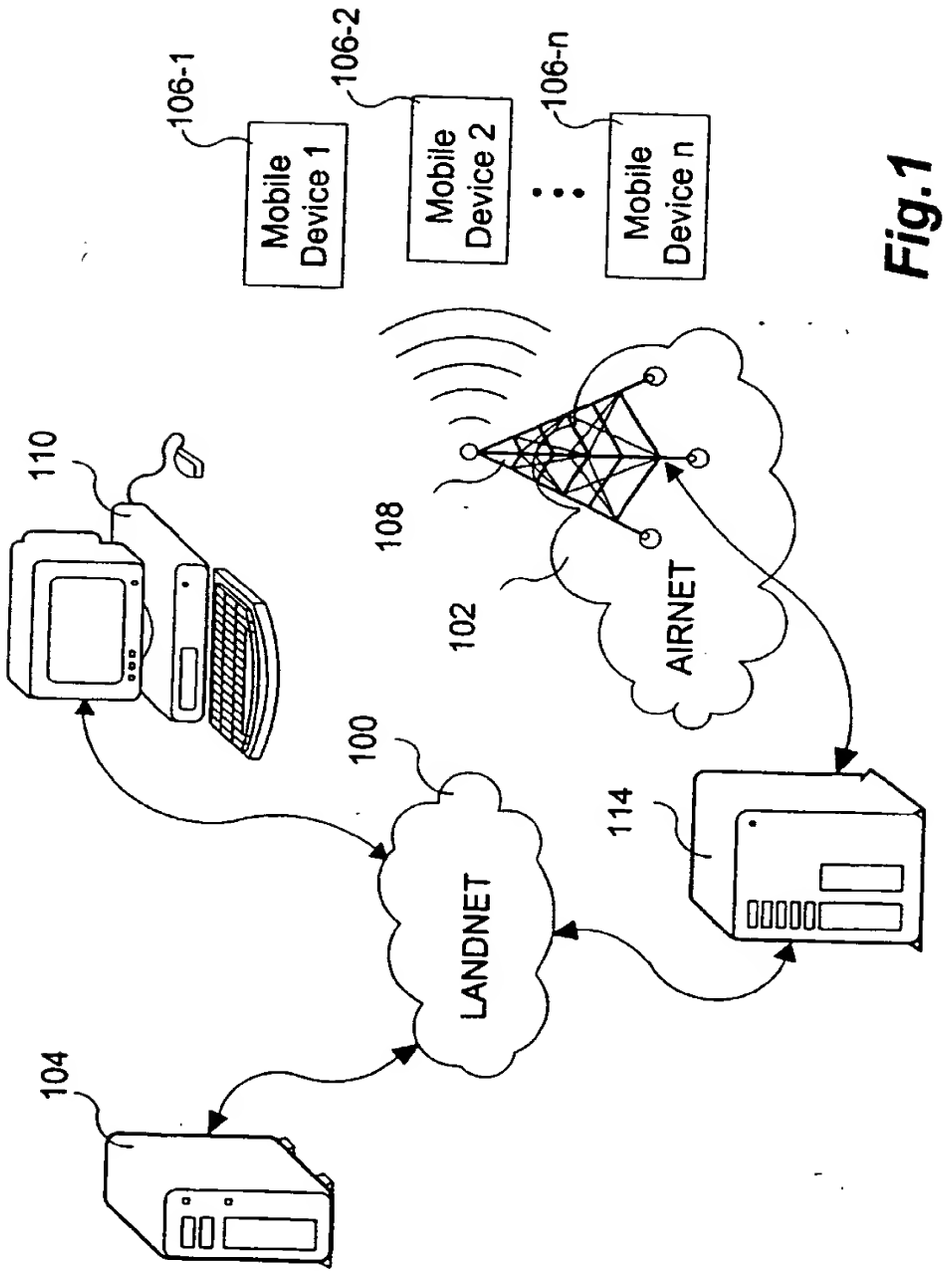
106-1
Mobile Device 1
106-2
Mobile Device 2
106-n
Mobile Device n
110
108
102
AIRNET
100
LANDNET
114
104

Fig.1

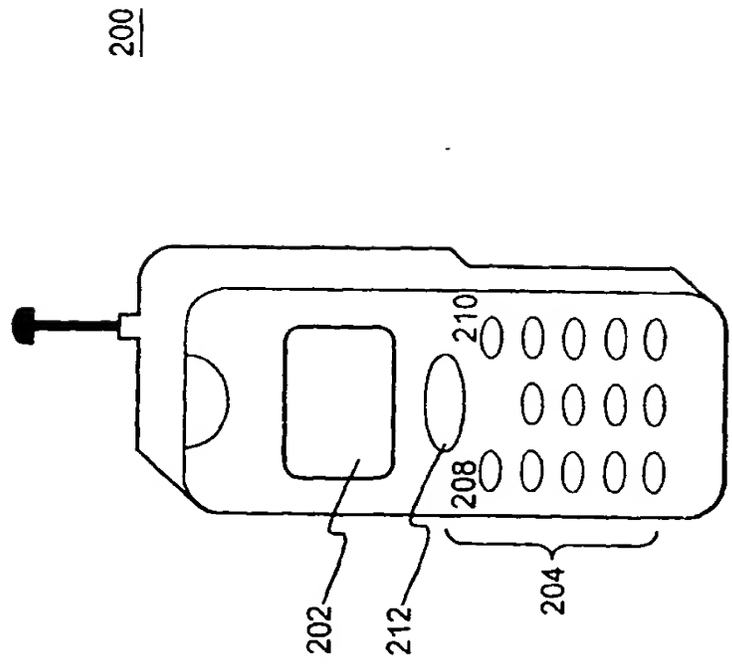

Fig. 2A

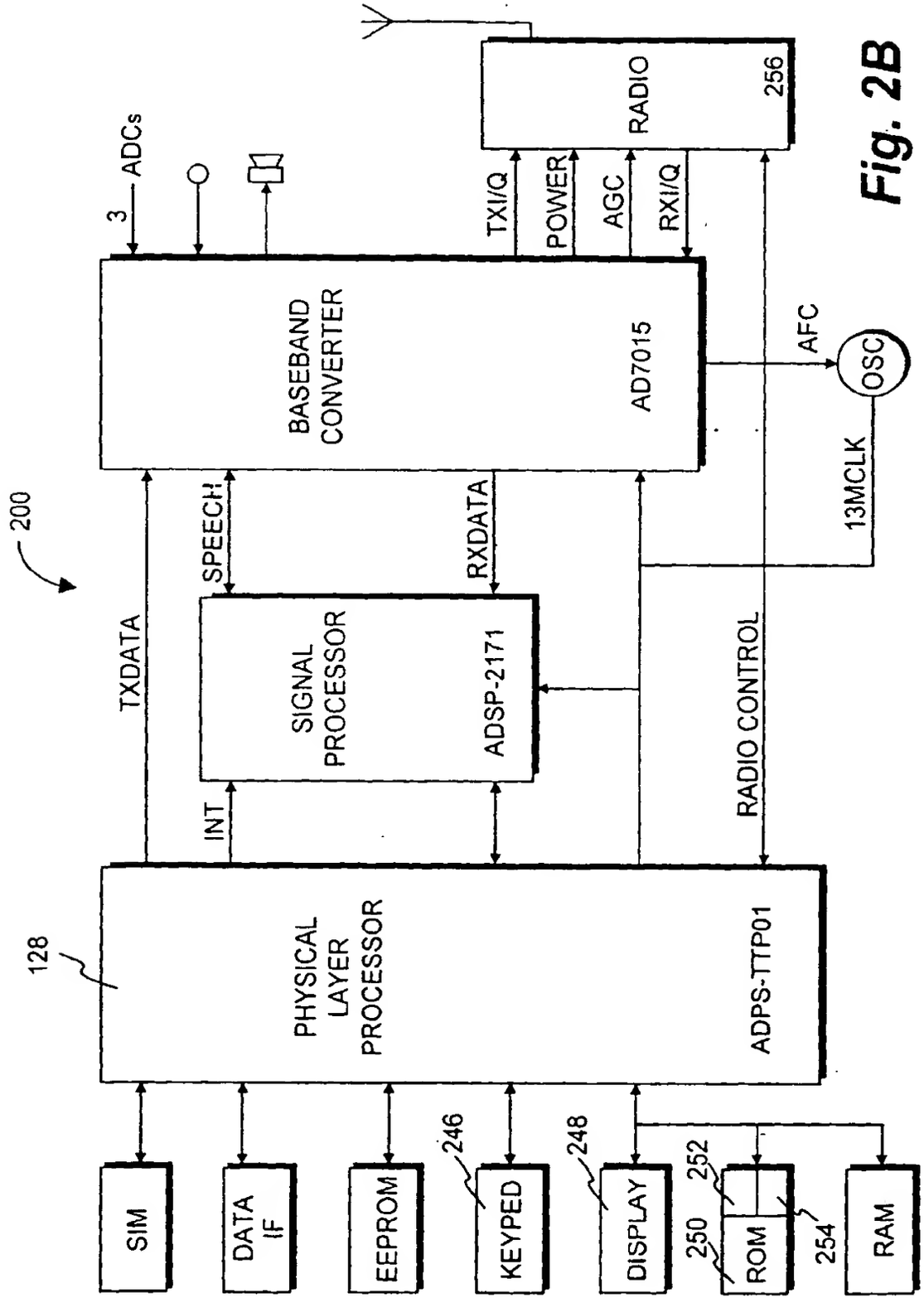
200
128
PHYSICAL LAYER PROCESSOR
ADPS-TTP01
BASEBAND CONVERTER
AD7015
RADIO
256
3 ADCs
TXI/Q
POWER
AGC
RXI/Q
AFC
OSC
13MCLK
SPEECH
RXDATA
TXDATA
INT
SIGNAL PROCESSOR
ADSP-2171
RADIO CONTROL
SIM
DATA IF
EEPROM
KEYPED
246
DISPLAY
248
ROM
250
252
RAM
254

Fig. 2B

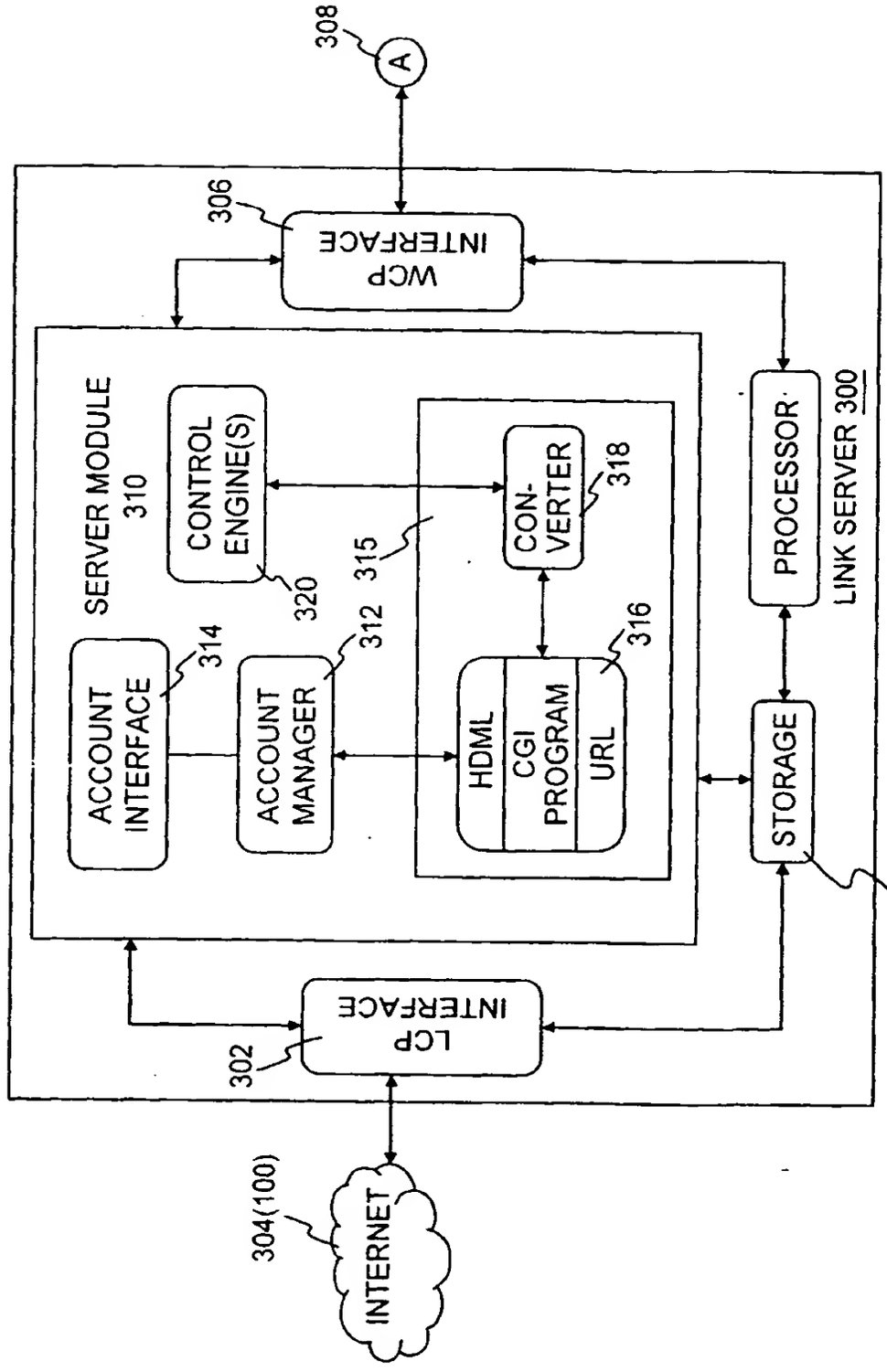
308
A
306
WCP INTERFACE
SERVER MODULE 310
CONTROL ENGINE(S)
320
315
CON-VERTER
318
ACCOUNT INTERFACE
314
ACCOUNT MANAGER
312
HDML
CGI PROGRAM
URL
316
PROCESSOR
LINK SERVER 300
STORAGE
316
LCP INTERFACE
302
304(100)
INTERNET

Fig. 3A

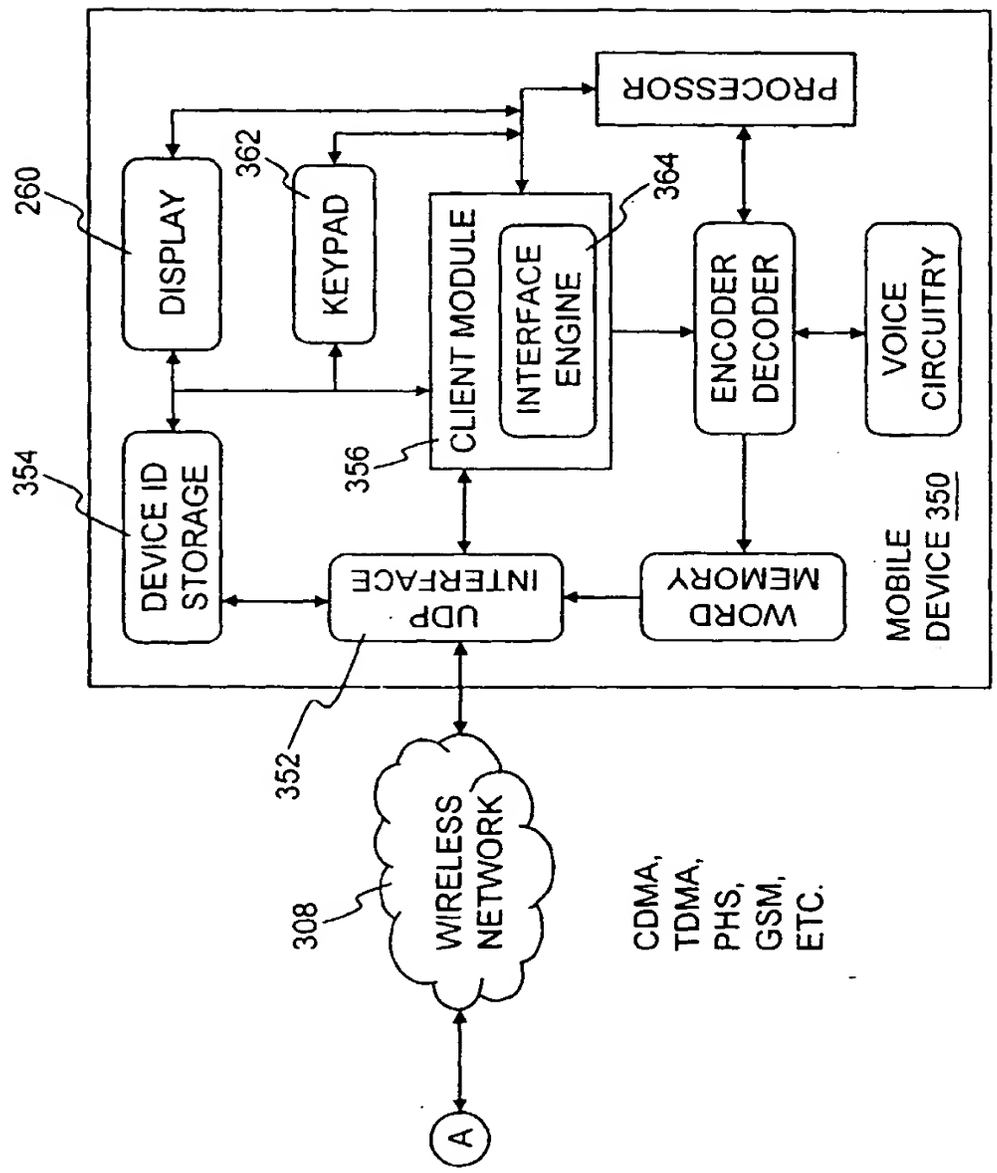
PROCESSOR
260
DISPLAY
362
KEYPAD
364
CLIENT MODULE
INTERFACE ENGINE
ENCODER DECODER
VOICE CIRCUITRY
354
DEVICE ID STORAGE
356
UDP INTERFACE
WORD MEMORY
MOBILE DEVICE 350
352
WIRELESS NETWORK
308
CDMA,
TDMA,
PHS,
GSM,
ETC.
A

Fig. 3B

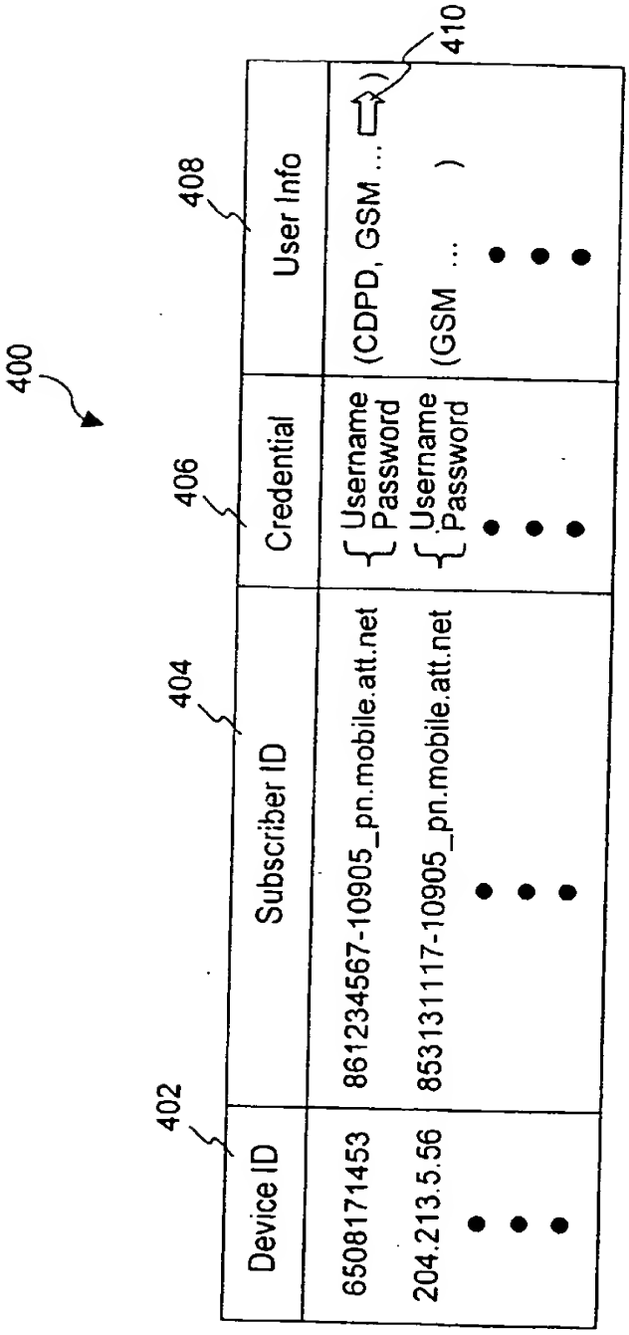
400
402
404
406
408
410
Device ID
Subscriber ID
Credential
User Info
6508171453
861234567-10905_pn.mobile.att.net
Username
Password
(CDPD, GSM ...)
204.213.5.56
853131117-10905_pn.mobile.att.net
Username
Password
(GSM ...)

Fig. 4

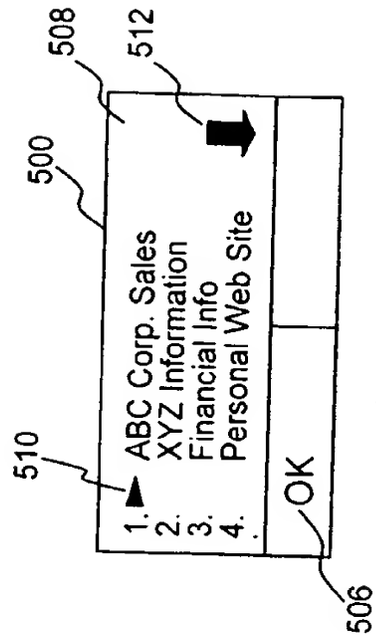
500
508
512
510
1. ABC Corp. Sales
2. XYZ Information
3. Financial Info
4. Personal Web Site
OK
506

Fig. 5B

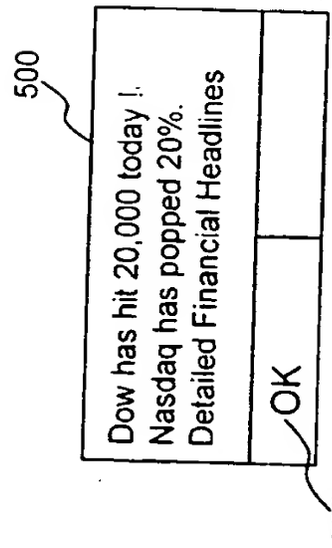
500
Dow has hit 20,000 today !
Nasdaq has popped 20%.
Detailed Financial Headlines
OK
506

Fig. 5A

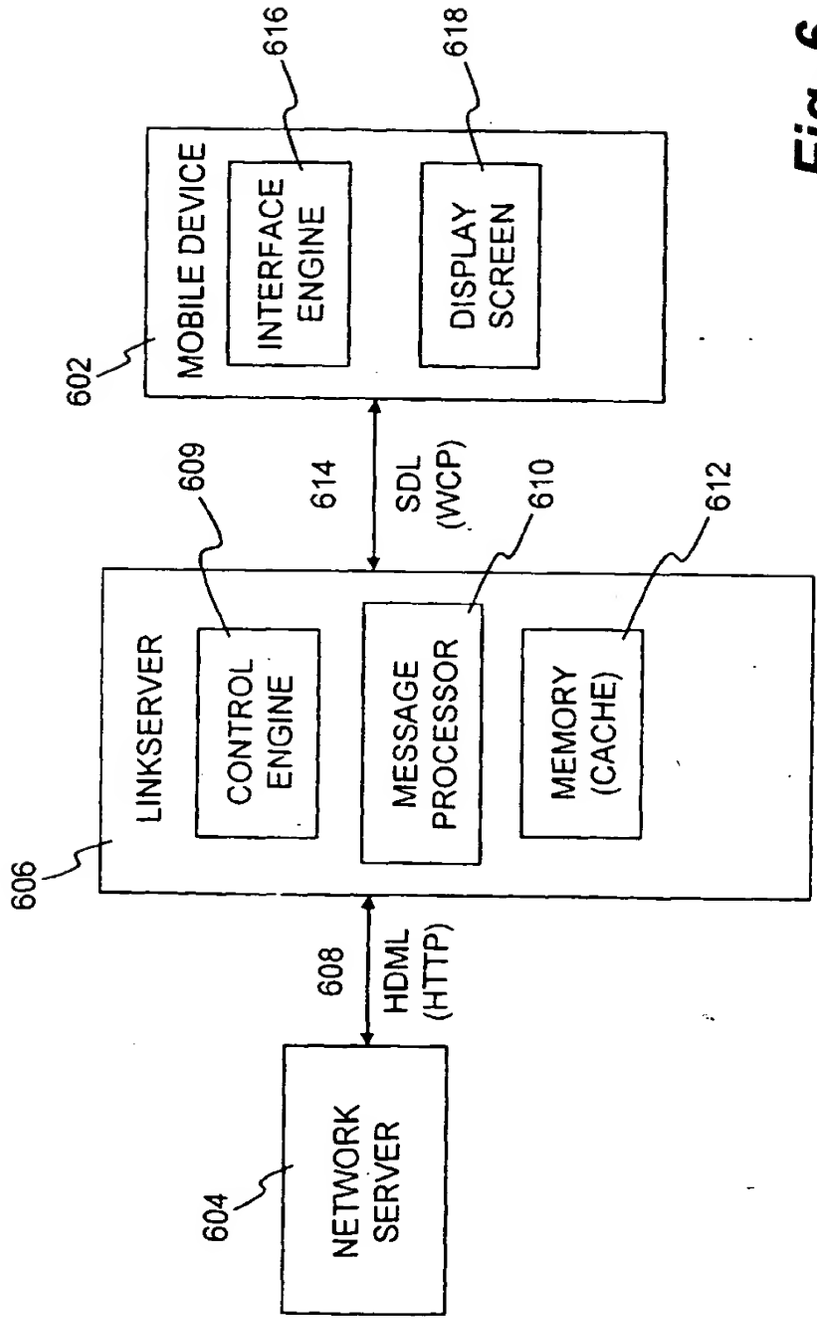
MOBILE DEVICE
INTERFACE ENGINE
DISPLAY SCREEN
602
616
618
614
SDL (WCP)
609
610
612
LINKSERVER
CONTROL ENGINE
MESSAGE PROCESSOR
MEMORY (CACHE)
606
608
HDML (HTTP)
NETWORK SERVER
604

Fig. 6

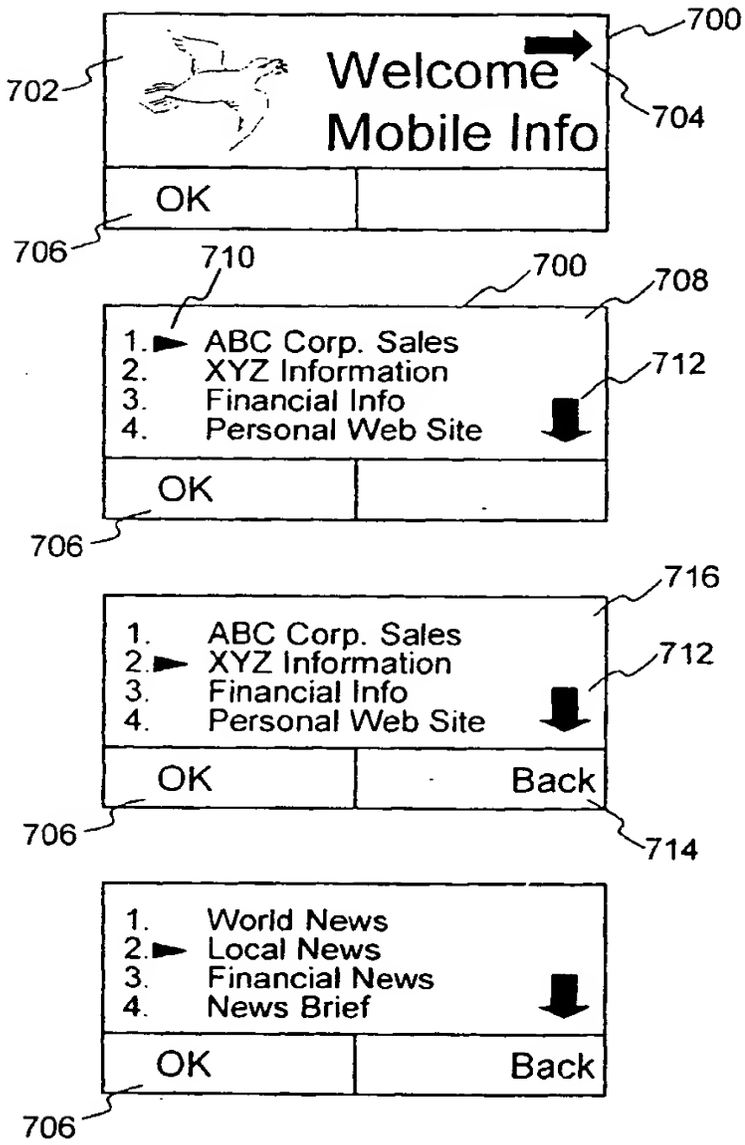

Fig. 7A

Fig. 7B

Fig. 7C

Fig. 7D

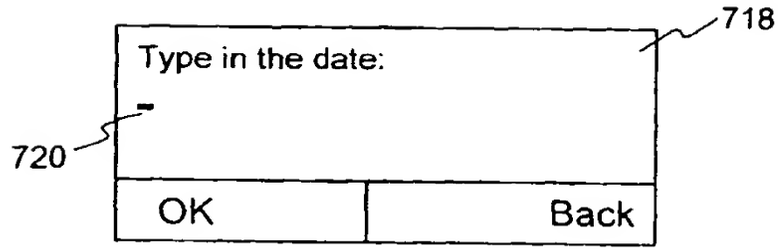

Fig. 7E

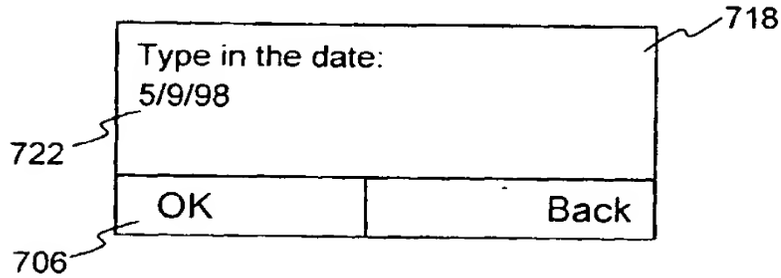

Fig. 7F

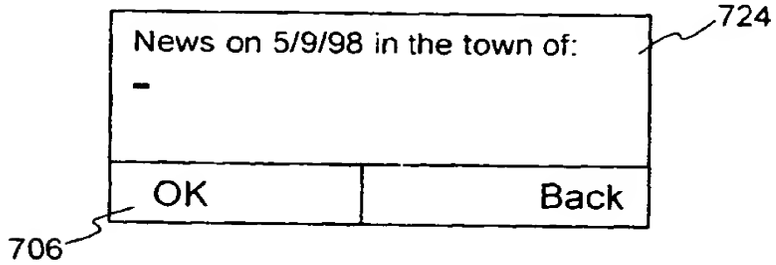

Fig. 7G

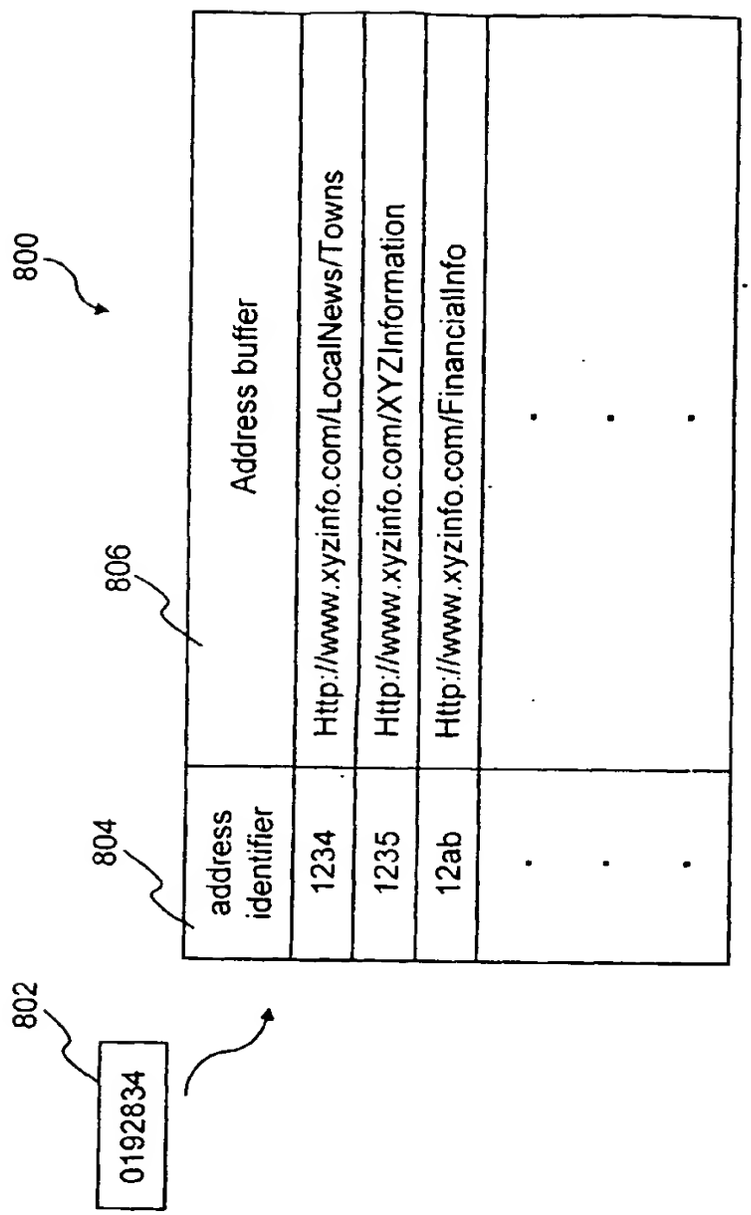
800
806
804
802
address identifier
Address buffer
1234
Http://www.xyzinfo.com/LocalNews/Towns
1235
Http://www.xyzinfo.com/XYZInformation
12ab
Http://www.xyzinfo.com/FinancialInfo
0192834

Fig. 8A

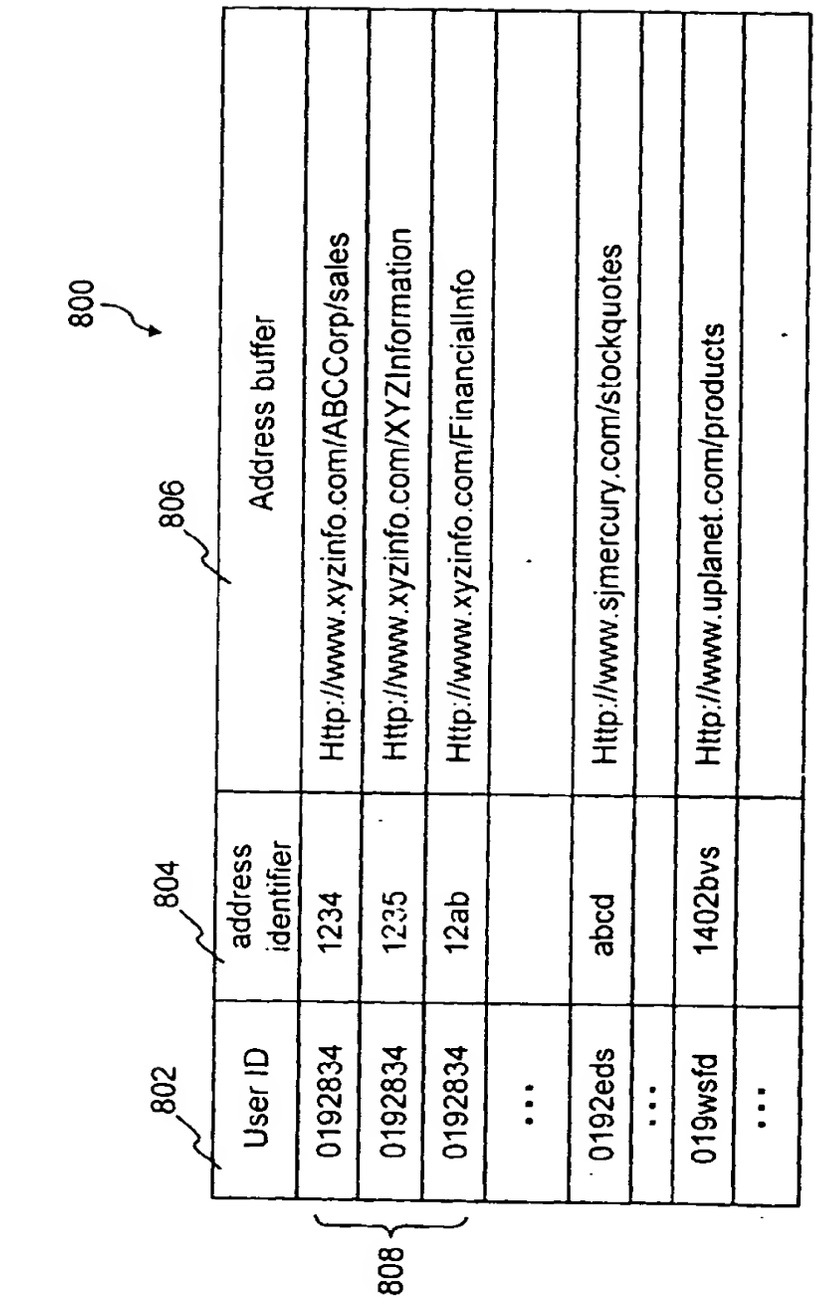
800
802
804
806
808
User ID
address identifier
Address buffer
0192834
1234
Http://www.xyzinfo.com/ABCCorp/sales
0192834
1235
Http://www.xyzinfo.com/XYZInformation
0192834
12ab
Http://www.xyzinfo.com/FinancialInfo
...
0192eds
abcd
Http://www.sjmercury.com/stockquotes
...
019wsfd
1402bvs
Http://www.uplanet.com/products
...

Fig. 8B

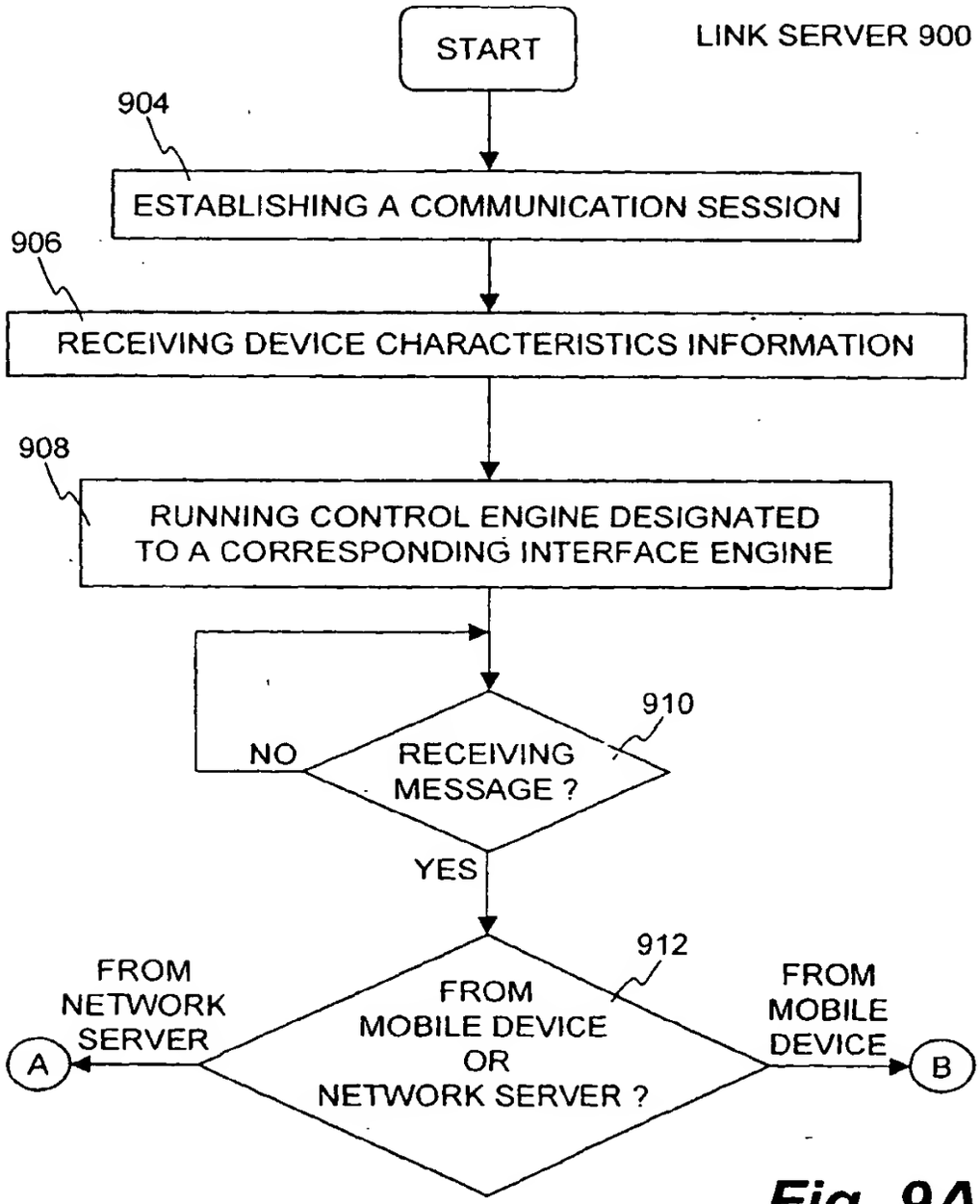
START
LINK SERVER 900
904
ESTABLISHING A COMMUNICATION SESSION
906
RECEIVING DEVICE CHARACTERISTICS INFORMATION
908
RUNNING CONTROL ENGINE DESIGNATED TO A CORRESPONDING INTERFACE ENGINE
910
NO
RECEIVING MESSAGE ?
YES
912
FROM NETWORK SERVER
FROM MOBILE DEVICE OR NETWORK SERVER ?
FROM MOBILE DEVICE
A
B

Fig. 9A

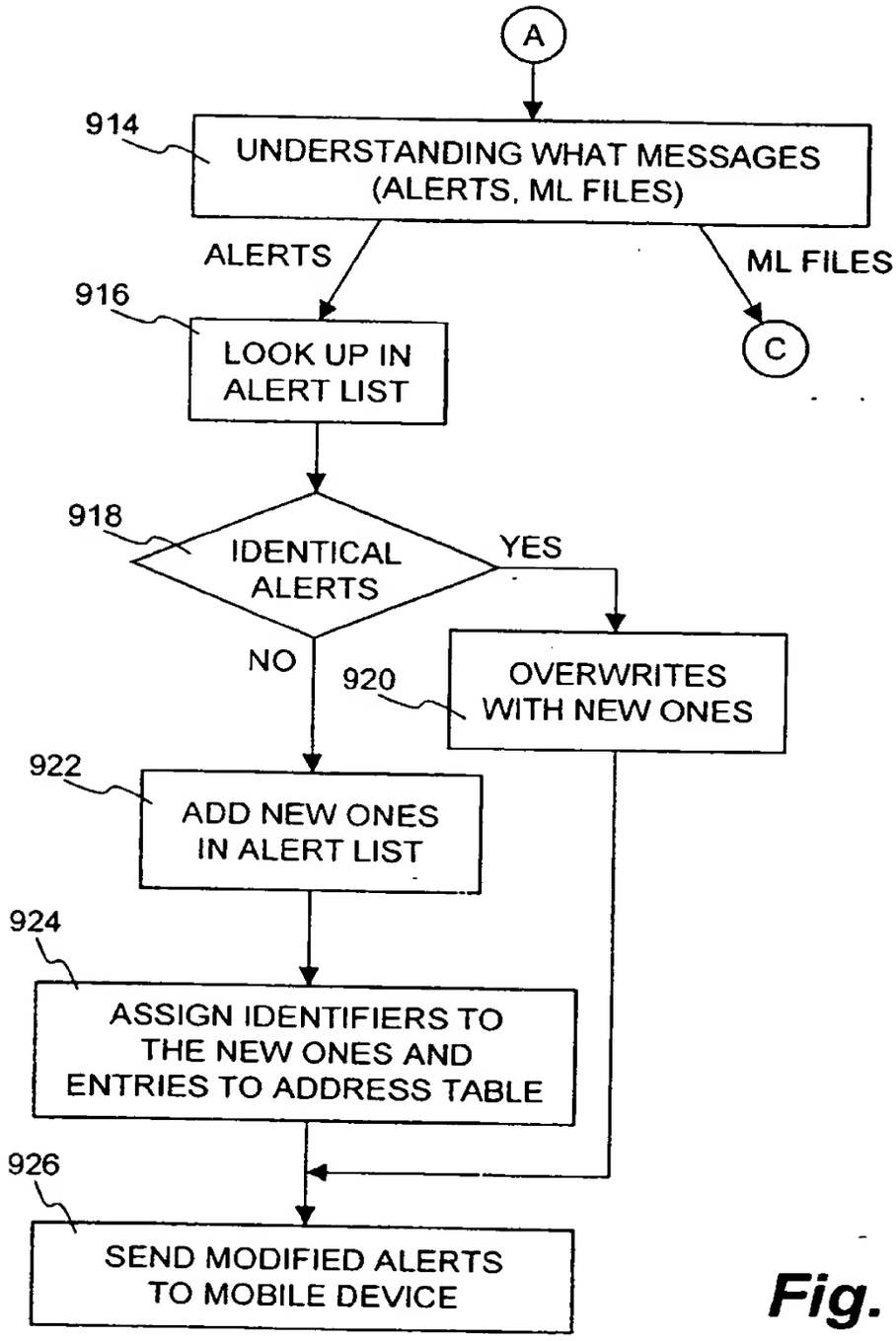
A
914
UNDERSTANDING WHAT MESSAGES (ALERTS, ML FILES)
ALERTS
ML FILES
916
LOOK UP IN ALERT LIST
C
918
IDENTICAL ALERTS
YES
NO
920
OVERWRITES WITH NEW ONES
922
ADD NEW ONES IN ALERT LIST
924
ASSIGN IDENTIFIERS TO THE NEW ONES AND ENTRIES TO ADDRESS TABLE
926
SEND MODIFIED ALERTS TO MOBILE DEVICE

Fig. 9B

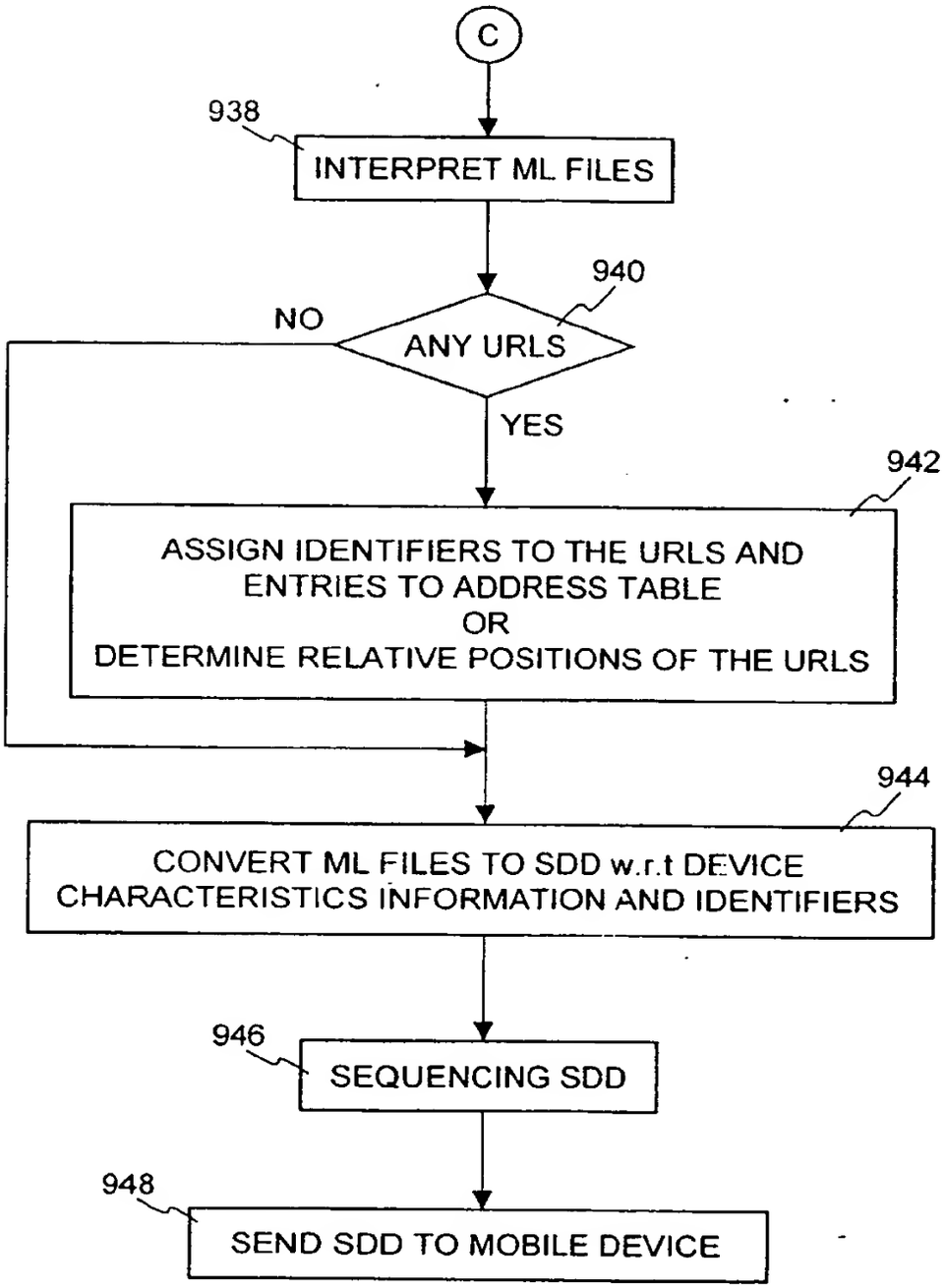
C
938
INTERPRET ML FILES
940
NO
ANY URLS
YES
942
ASSIGN IDENTIFIERS TO THE URLS AND ENTRIES TO ADDRESS TABLE OR DETERMINE RELATIVE POSITIONS OF THE URLS
944
CONVERT ML FILES TO SDD w.r.t DEVICE CHARACTERISTICS INFORMATION AND IDENTIFIERS
946
SEQUENCING SDD
948
SEND SDD TO MOBILE DEVICE

Fig. 9C

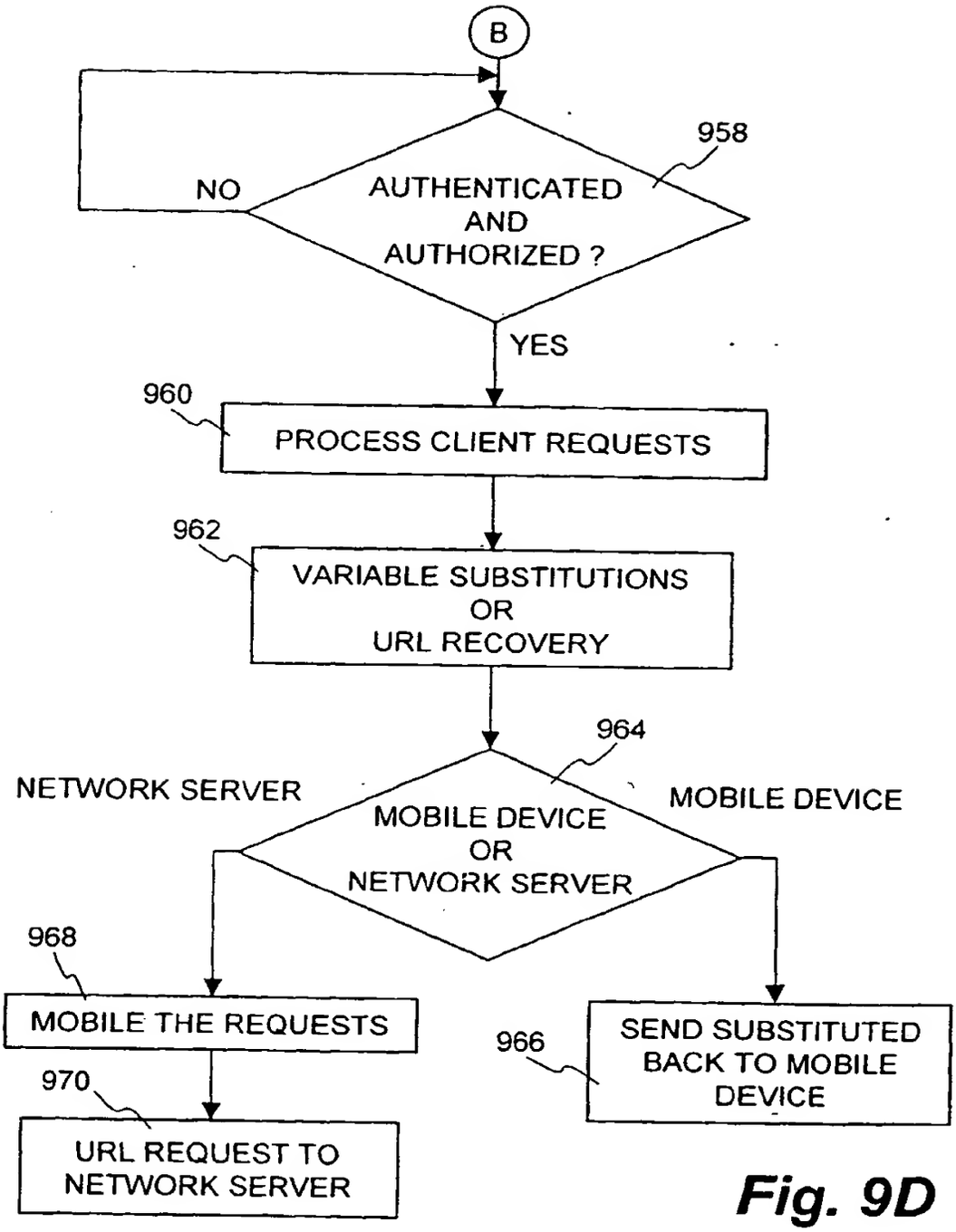
B
958
NO
AUTHENTICATED AND AUTHORIZED ?
YES
960
PROCESS CLIENT REQUESTS
962
VARIABLE SUBSTITUTIONS OR URL RECOVERY
964
NETWORK SERVER
MOBILE DEVICE
MOBILE DEVICE OR NETWORK SERVER
968
MOBILE THE REQUESTS
966
SEND SUBSTITUTED BACK TO MOBILE DEVICE
970
URL REQUEST TO NETWORK SERVER

Fig. 9D

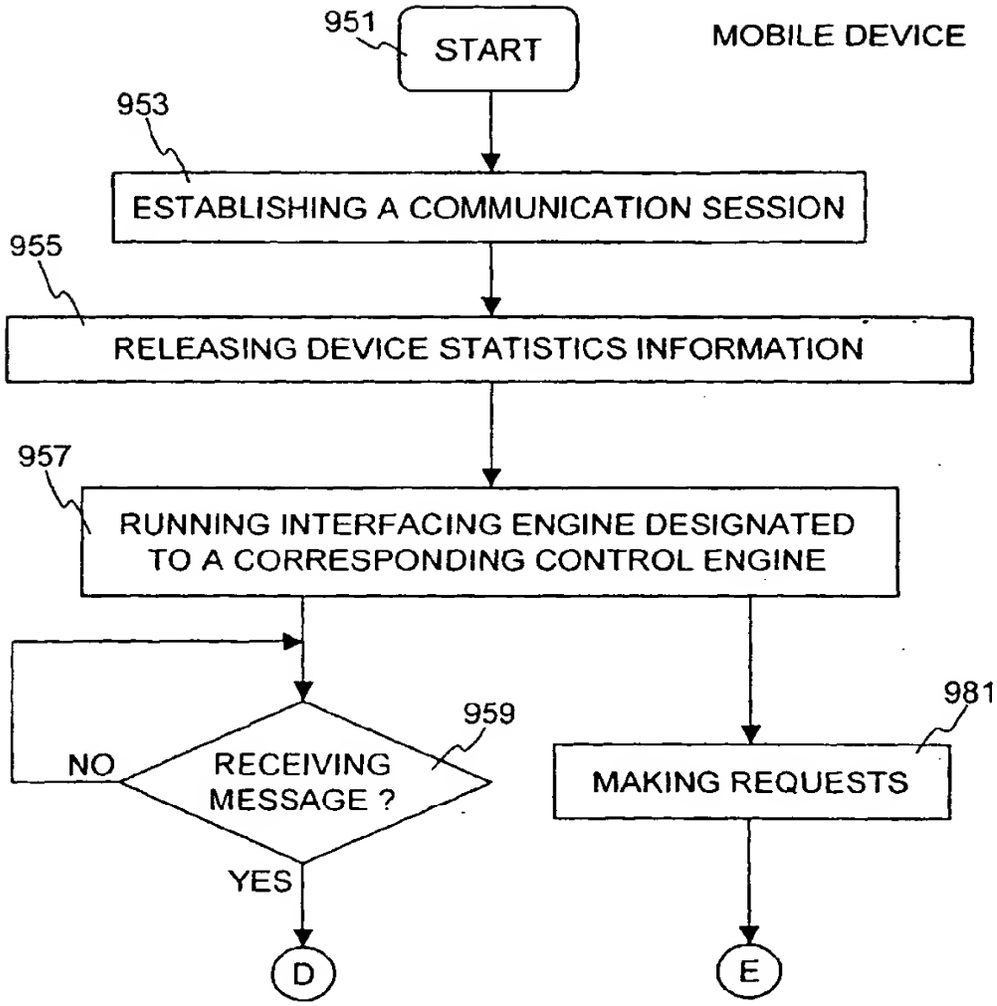
MOBILE DEVICE
951
START
953
ESTABLISHING A COMMUNICATION SESSION
955
RELEASING DEVICE STATISTICS INFORMATION
957
RUNNING INTERFACING ENGINE DESIGNATED TO A CORRESPONDING CONTROL ENGINE
959
RECEIVING MESSAGE ?
NO
YES
D
981
MAKING REQUESTS
E

Fig. 9E

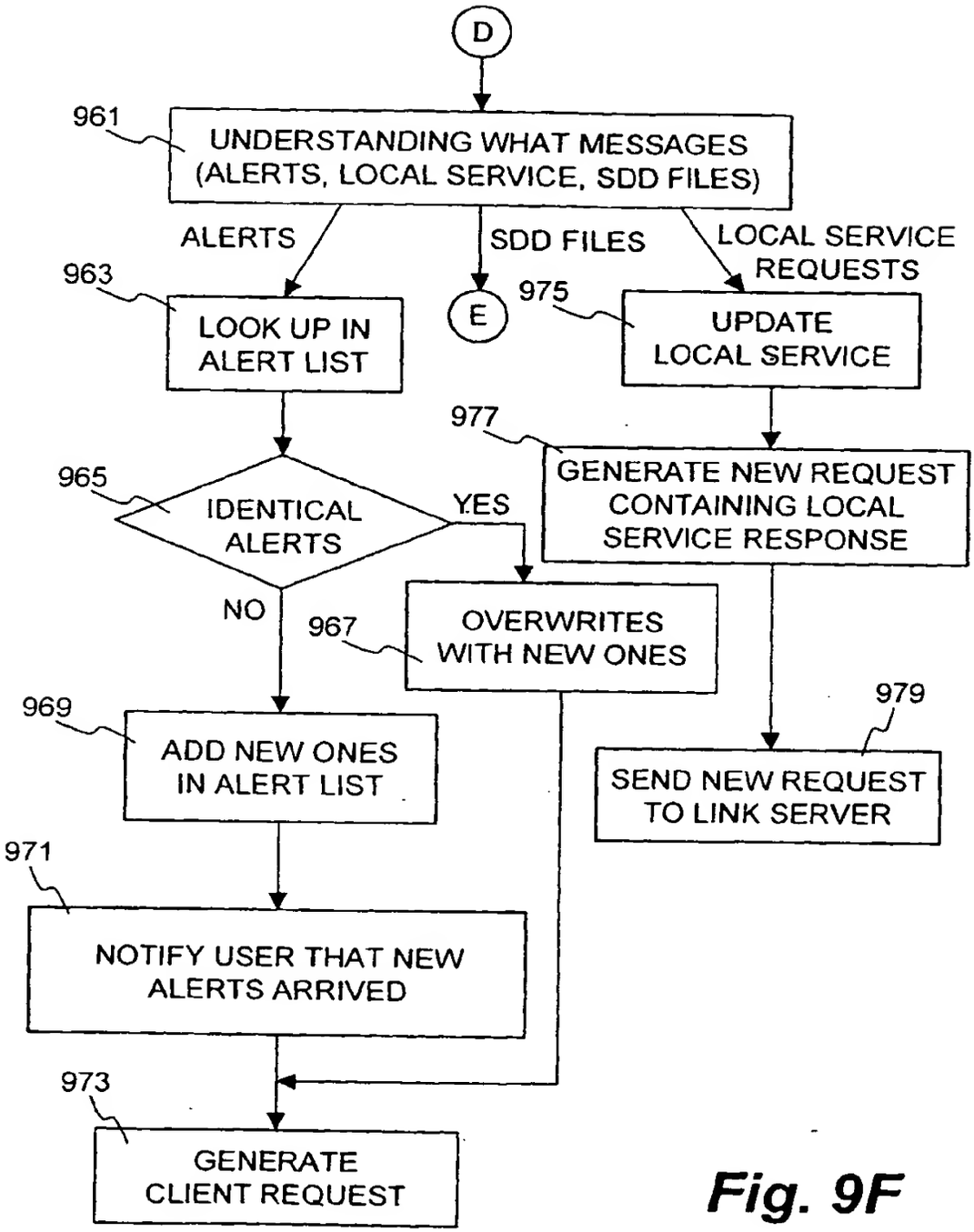
D
961
UNDERSTANDING WHAT MESSAGES (ALERTS, LOCAL SERVICE, SDD FILES)
ALERTS
SDD FILES
LOCAL SERVICE REQUESTS
963
LOOK UP IN ALERT LIST
E
975
UPDATE LOCAL SERVICE
977
GENERATE NEW REQUEST CONTAINING LOCAL SERVICE RESPONSE
965
IDENTICAL ALERTS
YES
NO
967
OVERWRITES WITH NEW ONES
969
ADD NEW ONES IN ALERT LIST
979
SEND NEW REQUEST TO LINK SERVER
971
NOTIFY USER THAT NEW ALERTS ARRIVED
973
GENERATE CLIENT REQUEST

Fig. 9F

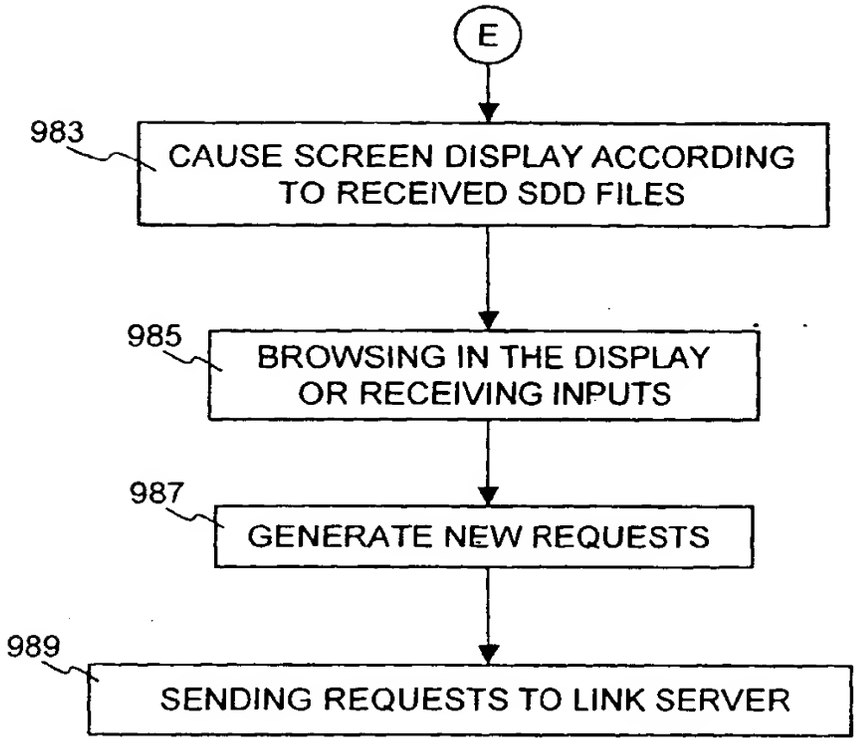
E
983
CAUSE SCREEN DISPLAY ACCORDING TO RECEIVED SDD FILES
985
BROWSING IN THE DISPLAY OR RECEIVING INPUTS
987
GENERATE NEW REQUESTS
989
SENDING REQUESTS TO LINK SERVER

Fig. 9G

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 987 868 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
23.05.2001 Bulletin 2001/21

(43) Date of publication A2:
22.03.2000 Bulletin 2000/12

(21) Application number: **99307294.1**

(22) Date of filing: **14.09.1999**

(51) Int Cl.[7]: **H04L 29/06**, H04Q 7/32

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **14.09.1998 US 153322**

(71) Applicant: **Phone.Com Inc.**
Redwood City, CA 94063 (US)

(72) Inventors:
- **Schwartz, Bruce V.**
  San Mateo, CA 94402 (US)
- **Greer, Russell S.**
  Los Gatos, CA 95030 (US)
- **Boyle, Stephen S.**
  Fremont, CA 94539 (US)
- **Fox, Mark A.**
  San Mateo, CA 94403 (US)
- **Rossmann, Alain S.**
  Palo Alto, CA 94303 (US)
- **Lentezner, Mark G.**
  Mountain View, CA 94041 (US)
- **Laursen, Andrew L.**
  San Mateo, CA 94402 (US)
- **Sandman, Brad E.**
  Sunnyvale, CA 94086 (US)

(74) Representative: **Ablett, Graham Keith et al**
Ablett & Stebbing,
Caparo House,
101-103 Baker Street
London W1M 1FD (GB)

(54) Method and architecture for interactive two-way communication devices to interact with a network

(57) The present invention relates to navigation of the Internet by a two-way interactive communication mobile device capable of wireless communication, via a link server (300), with service providers or network servers on the Internet. After the mobile device has established a communication session with the link server over a wireless network (308), a control engine 320 in the link server is initiated and uses the computing resources of the link server device so as to be responsible for tasks that require considerable computing power and memory. such as processing of URL requests, interpretation of markup language files, management of data cache and variable states. Working with a message processor (315) in the server device, the control engine communicates with an interface engine in the mobile device using a compact data format that is efficiently transportable in the wireless data network. The interface engine typically performs tasks that do not require considerable computing power and memory, such as receiving input data from users, and the rendering of the compact data format received from the link server device, to cause the mobile device to display contents in the markup language files on a display screen.

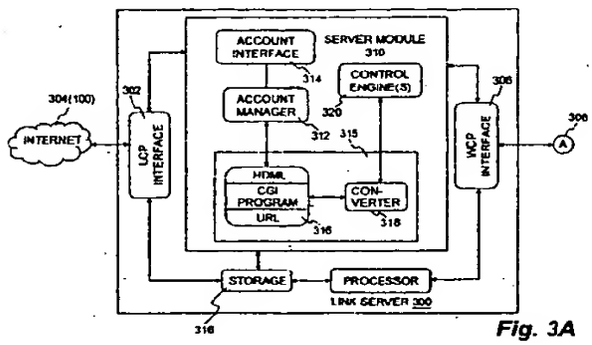

Fig. 3A

EP 0 987 868 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 99 30 7294

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	MEYER M ET AL: "THE ON-THE-MOVE CONCEPT FOR MOBILE MIDDLEWARE" ISS. WORLD TELECOMMUNICATIONS CONGRESS. (INTERNATIONAL SWITCHING SYMPOSIUM),CA,TORONTO, PINNACLE GROUP, 21 September 1997 (1997-09-21), pages 373-378, XP000704489	1-3,6,7, 9,10, 12-14	H04L29/06 H04Q7/32
Y	* abstract *	4,8,11	
A	* page 374, left-hand column, line 4 - page 377, left-hand column, line 11 *	5	
Y	WO 97 27546 A (EX MACHINA INC) 31 July 1997 (1997-07-31)	4,8,11	
A	* abstract * * page 1, line 10 - page 5, line 9 * * page 7, line 27 - page 15, line 16 * * page 17, line 17 - page 18, line 26 * * page 31, line 8-20 * * page 34, line 18 - page 35, line 4 * * page 36, line 20 - page 37, line 21 * * page 42, line 1 - page 50, line 9 * * figure 1 *	5	
X	US 5 727 159 A (KIKINIS DAN) 10 March 1998 (1998-03-10) * abstract * * column 1, line 1 - column 13, line 9 * * figures 1-5 *	1-3,6,7, 9,10, 12-14	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)
H04L
H04Q

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	29 March 2001	Lievens, K

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 99 30 7294

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

29-03-2001

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9727546 A	31-07-1997	AU 1754397 A	20-08-1997
		CA 2243555 A	31-07-1997
		CN 1217800 A	26-05-1999
		EP 0886826 A	30-12-1998
		US 6021433 A	01-02-2000
US 5727159 A	10-03-1998	CN 1218561 A	02-06-1999
		EP 0892947 A	27-01-1999
		JP 11508715 T	27-07-1999
		WO 9738389 A	16-10-1997
		US 6076109 A	13-06-2000

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

PERGAMON

Computers & Graphics 23 (1999) 841–848

COMPUTERS &GRAPHICS

www.elsevier.com/locate/cag

IMC '98

Resource adaptive WWW access for mobile applications

C. Freytag[a], L. Neumann[b,*]

[a]*Interactive Graphics System Group, Technical University Darmstadt (TUD), Rundeturmstr 6, 64283 Darmstadt, Germany*
[b]*Department of Mobile Information Visualization, Computer Graphics Center (ZGDV), Darmstadt, Germany*

Abstract

Mobile Computing and World Wide Web are rapidly gaining popularity. They are fundamental technologies for a vision of information access for anyone, anytime, anywhere. However, mobile information access has to cope with the heterogeneity, dynamic and restrictions (e.g., narrow bandwidth, restricted resources) of the mobile environment. In this paper we present a system that supports a useful integration of portable computing devices such as (Sub-)Notebook, Palmtop or Personal Digital Assistants within the present WWW infrastructure. We propose an architecture for a context-aware adaptation within a mobile WWW environment. It enables resource adaptive access to any content within the WWW. The main idea is to exploit meta-information about the resource environment and user preferences to find a suitable adaptation pipeline to overcome most limitations of mobile information access. We discuss our first experiences we made with AdaWeb prototype. Then we present the extensions of our adaptation system to improve the mobile WWW access.

Keywords: Mobile computing; World Wide Web; Adaptation; Proxy; PDA

1. Introduction

With the development of enabling technologies such as wireless networking and various types of affordable mobile devices, mobile computing has started to be widely accepted and applied. With this, the possibilities of mobile data communication using wireless networks are of rising importance. This mobile technology will also become more and more integrated into the World Wide Web in order to be able to access "everywhere" in the Web. The combination of mobile technology with WWW is one important step towards a global information infrastructure.

In general, a mobile WWW application requires a mobile device such as Personal Digital Assistants (PDAs) or notebooks to access a remote WWW server. The connection to a WWW-server can be established with communication facilities provided by a mobile phone or the mobile computing device itself (see Fig. 1). This enables mobile users equipped with a portable computer to access information anywhere and anytime. However, these kind of applications face new challenges that are mainly caused by the limited resources (e.g., display, battery, memory size, processing power) of mobile devices compared to desktop environments and the low bandwidth of wireless narrowband wide-area networks. Due to these shortcomings, new concepts are needed to provide efficient access to WWW servers within a mobile environment. Otherwise, WWW applications will be rejected by the users. In this context, adaptation is one of the most important concepts, firstly, in overcoming the most serious problems and secondly, in providing an efficient access to information services within the limits of the system.

Adaptation can be achieved at various levels (e.g., network-, system-, application-level) within a mobile system and its applications. Several techniques, such as pre-fetching, caching, compression, and filtering are suitable for this purpose. These techniques can be applied to utilise the bandwidth efficiently, reduce communication overhead, and hide network latency.

We have developed an adaptation concept at application level applying a rule-based system that utilises the

* Corresponding author. Tel.: +49-6151-155-617.
E-mail address: neumann@zgdv.de (L. Neumann)

0097-8493/99/$ - see front matter
PII: S0097-8493(99)00114-4

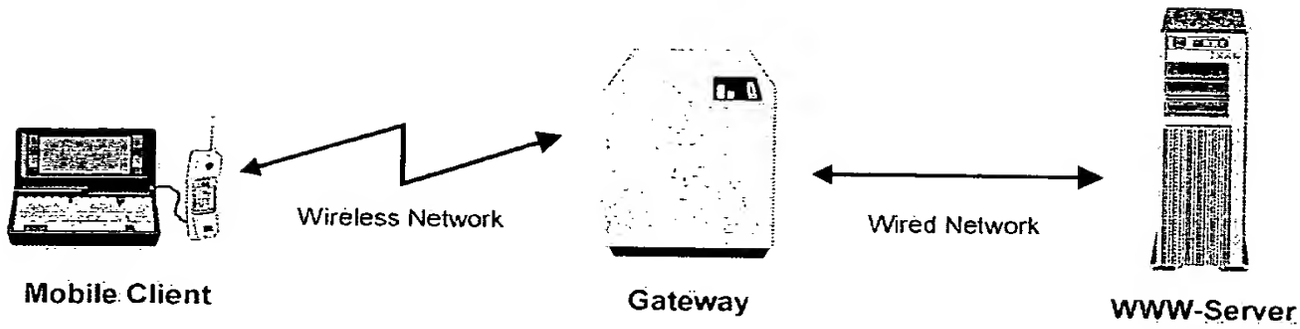

Fig. 1. Mobile WWW application scenario.

knowledge of the resource environment and user preferences to adjust the access of distributed information services and presentation within the limits of the mobile system. In this paper, we will give an overview of our work by introducing the system architecture and its main components. Then, we will focus on the management of resources with regard to mobile WWW access. The resource management encompasses the functionality for the maintenance of resource information such as storage, retrieval, deletion and update of resource characteristics, and to monitor resources. Then, we will introduce the trader component as a component that helps to keep the latency of the adaptation service low and we will also discuss the introduction of client components to enhance the adaptation process (data compression, location tracking).

2. Adaptation concepts for a mobile WWW access

Looking at present the WWW technology with regard to its applicability for a dynamic content adaptation, we can identify the following aspects:

- Common approaches providing dynamic adaptation of WWW content are techniques using cookies, URL extensions or CGI scripts. However, these solutions do not support the specification and exploitation of user preferences or resource profiles for a context-aware adaptation.
- The current version of HTML offers some tags and attributes like ALT to enable the presentation of an alternative content. However, HTML is not designed for the definition of different layouts for various screen resolutions.
- When using Java or JavaScript for the adaptation, the creation of HTML pages would be extremely expensive due to the necessity of specific knowledge of the developer. Furthermore, the original WWW content would be transferred to the client and the adaptation would use the resources on the client side and require Java/JavaScript support.

To summarise, all approaches mentioned above do not provide sufficient functionality for a dynamic adaptation. Therefore, several proposals address the theme of mobile WWW access by using an application-level approach. In general, they adjust the WWW content presentation to the limits of the specific mobile device and the mobile connection. One possible model is the adaptation of WWW content on the client side. This is often used by WWW browser developers. However, due to the restricted memory and the low processing power, a mobile client (e.g., a PDA) will not allow expensive computing adaptation operations. Moreover, if it is not possible to present the WWW content on the client device and it is therefore filtered out, then the data has been transferred unnecessarily.

A further approach is to use machines within the fixed network, that pre-process or filter the WWW content for the mobile device facilities. This can be done on the server itself or on an intermediate site, a so-called proxy between client and server. Using a proxy for the WWW access is a common approach [1–3]. However, the current solutions are not generic in terms of hardware profiles and user preferences. Therefore, we propose a dynamic WWW content adaptation system that evaluates user preferences and available resources to identify the most suitable adaptation strategy for the current context.

3. Related work

At present, mobile WWW access is being discussed by the W3-consortium. The *Mobile Access Interest Group* is endeavouring to create a proposal for a new WWW standard which supports resource-limited devices (e.g., mobile phone with text display only). Two proposals exist:

- Compact HTML [4] is a subset of different HTML standards. The selected tags and attributes are related to resource constraints of mobile clients.
- Wireless Mark-up Language (WML) [5] is the pendant to HTML. It is based on the Extended Markup Language (XML) and has been designed by the WAP-Group (Wireless Application Protocol).

These are activities that introduce new languages and proprietary protocols to improve the mobile WWW access. A widespread example is the popular TopGun WingMan for the Palm-computing platform [6]. There a *split-browser* is used: HTML parsing and conversion into a proprietary format is done on the proxy, displaying of this format is done by the browser. As already mentioned above, another approach is the use of proxies to adapt the access of information services. Within the WWW infrastructure various proxy configurations are possible. A proxy can be located on the client, on an intermediate site or even on the server itself. In general, one can distinguish between single- and dual-proxy architectures supporting mobile WWW applications [7].

TeleWeb [8] is one example for a single-proxy architecture, where the proxy is located on the client. It caches documents and enables an asynchronous email style browsing that can be controlled by the user. But, this solution uses additional resources of the mobile computing device.

A number of applications use a dual-proxy architecture running a proxy on the client as well as in the fixed network. Then, additional resources of hosts located in the fixed network are used to pre-process the requested data for the mobile client. These tasks are performed on specific stationary servers that can co-operate with the proxies on the mobile clients. In other words, the application is split into two parts, where one part of the application complexity is moved to the fixed network. Typical examples for this category are Mowgli [9] and MobiScape [10]. They make use of a proxy on a mobile client and on a stationary server applying a cache on both sides. The WIT-project [11] uses proxies to split an application between client and server. However, they all need additional resources or functionality on the client side.

TranSend [12] and Digestor [13] are single-proxy architectures where the proxy is located in the fixed network. Digestor has an intelligent proxy that transforms Web-pages according to design heuristics. In order to reduce the bandwidth requirements, TranSend applies an infrastructure proxy service to provide datatype-specific distillation (i.e., quality reduction on images). Both systems need only standard Web browsers and move the computational complexity of the WWW access partly to the proxy within the fixed network.

All of these proposals provide solutions to enhance mobile WWW access, but they do not address dynamic adaptation of WWW content on different mobile clients. Furthermore, they do not answer the question of how already existing WWW content can be adapted for presentation on resource-limited mobile computing devices.

4. AdaWeb

The before-mentioned approaches have shown that a proxy approach offers the most advantages and should be preferred to the server or client approach. However, the above-listed proxy approaches are dedicated to only one or two special mobile devices and do not support a broad spectrum of mobile devices. Therefore, we have designed an adaptation service, that is placed on a proxy between client and server, and can adapt the WWW content, which a mobile user is able to access from the network, to the current context of the user. This context includes the mobile device, the user's preferences and the current available bandwidth of data transfer between the mobile device and the proxy. The concept of AdaWeb is shown in Fig. 2.

In order to carry out the adaptation, we are building a collection of adaptation rules. The rules define the transformation or modification of WWW content depending on the context parameters. These parameters have an influence on the premise of a rule, which is defined by a Boolean expression. When the premise is evaluated and the condition is fulfilled the actions of a rule are executed. An action correlates to one internal or external program that is responsible for a certain adaptation process. In the following paragraphs these kind of programs will be called *filters*.

The concept and architecture of AdaWeb has already been introduced in [14], but, in the meantime, the concept and the architecture have been extended for the integration of new components. In our previous concept, the adaptation was the only optimization of the WWW access and the whole adaptation process was localised on the proxy. The advantage of this is that no client component has to be developed for a mobile client. The development of a client component (e.g., browser, message handler) for a broad spectrum of mobile devices is no trivial task and is associated with high costs due to the different operating systems, the different programming

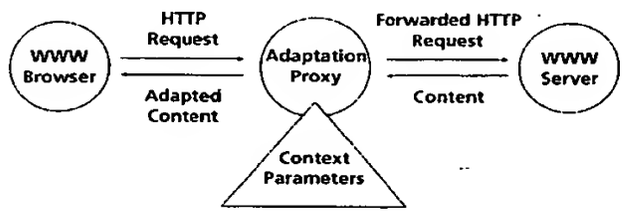

Fig. 2. Proxy-based content adaptation.

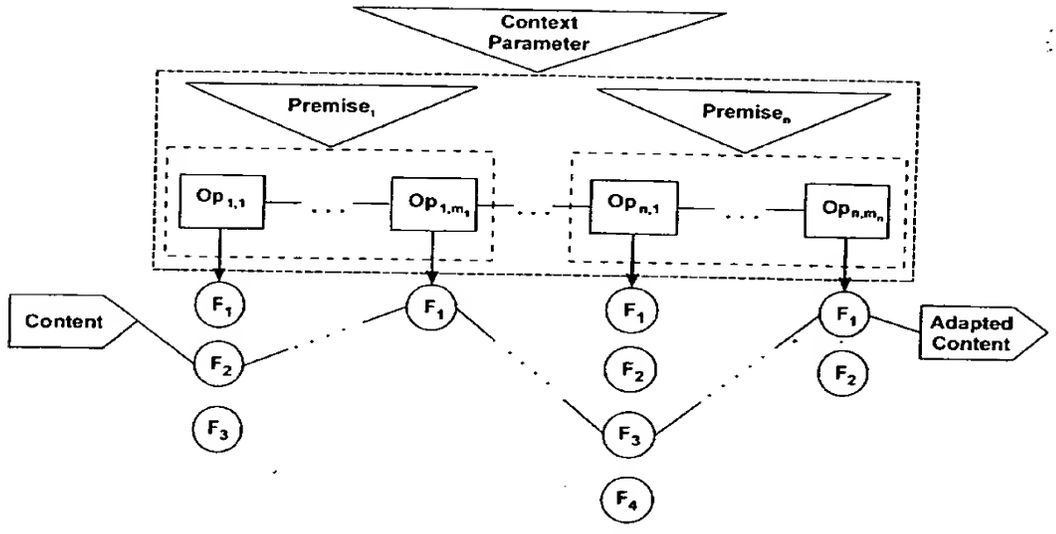

Fig. 3. Context-dependent adaptation pipeline with filter selection.

models and different developer tools. On the other side, content adaptation was the only technique that was supported by AdaWeb though there are a lot of other possibilities to make the mobile WWW access more efficient. To integrate approaches like a Level of Detail model for image transmission presented in [15] and a message handler according to this model using data compression and caching strategies [16], we have extended the concept and can also support a client-side component.

Another modification of our system relates to the regarded resources. In contrast to our previous approach, where the idea of a resource was identical to the idea of the mobile device, filters are managed as resources. Instead of calling up a filter directly within the action part of a rule, the name of the operation appears in this place. The name of the operation is a key to a list of filters which perform this operation (see Fig. 3). From this list the most appropriate filter is chosen, in general, the filter with the least workload.

4.1. Base system of AdaWeb

Fig. 4 shows the main components of the AdaWeb system. The highlighted components symbolise the extensions to the previous approach and are responsible for the realisation of the new concepts mentioned at the beginning of Section 4. In this section we will give a description of the components of the base system and discuss our first experiences.

Implemented components of AdaWeb which build the system according to the base system are:

- *Resource information base (RIB)* contains all information needed by AdaWeb, both from and about the outside world, and is therefore the central information interchange and storage management component of AdaWeb. It is used for user and session management and serves the role of interprocess communication.
- *User registration* is responsible for the administration of user accounts and user profiles. Device and user profiles are indicated by one key, which is transmitted to the user device as the result of the registration process.
- *Request manager* identifies the context of the web-client by analyzing cookie information or stripping special URL-prefixes. It then hands the request over to a normal caching proxy for retrieving the requested data from the foreign host. In the mean time it triggers the RIB to pre-load the data which defines the web-client context.
- *Adaptation component* loads the user and device information from the RIB and evaluates the premises of the adaptation rules. For every valid premise the operations of the corresponding action part are executed.

More detailed information about the listed parts of the system are given in [14]. These components represent the first prototype of AdaWeb which was used to evaluate our adaptation concept. Analyzing Web statistics with regard to the transferred volume of data has shown that images have a dominant percentage (73.5%) in contrast to HTML documents (23.5%). Filters for image operations like scaling and color reduction as well as format conversion are widely available, for example GIMP [17]. The definition of rules for the control of these filters

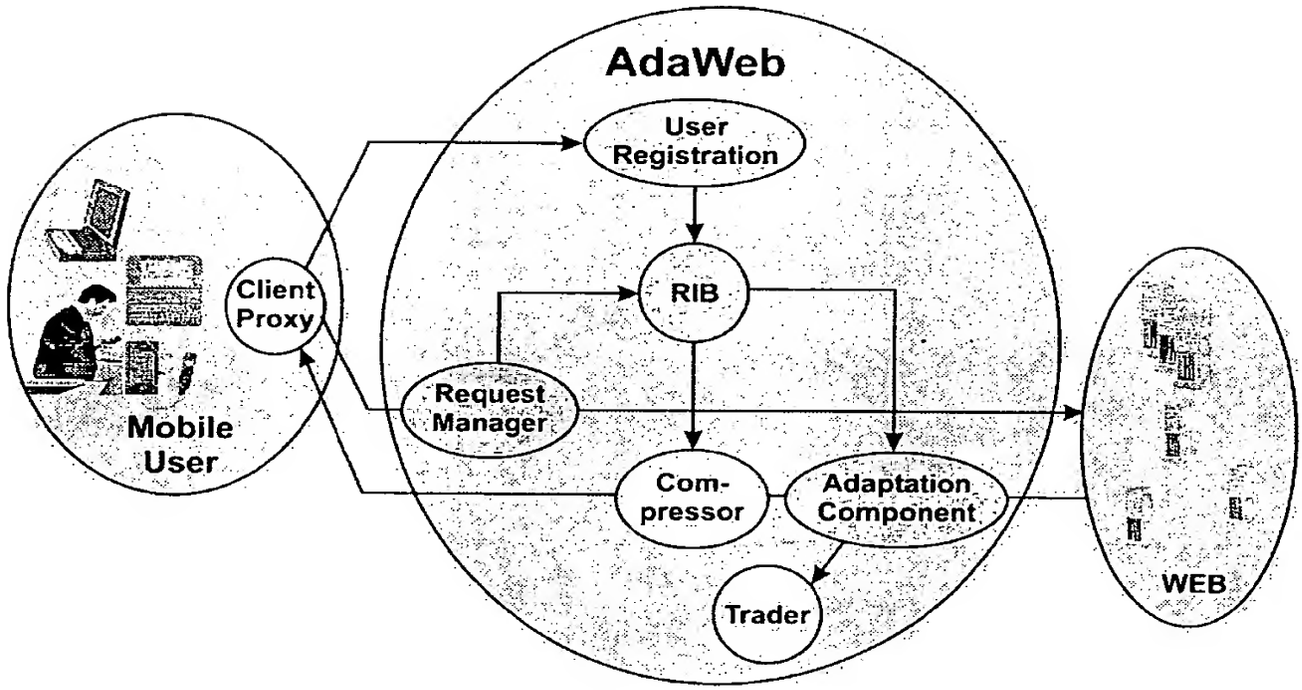

Fig. 4. Architecture of AdaWeb.

depending on display size and color depth of the mobile device or the network capability is not complex. Therefore, the introduction of image adaptation processes within our system is relatively simple and then by using AdaWeb transfer time could be significantly reduced by 60%.

The difficulties arise when trying to adapt HTML documents, because semantic information is required about the content as well as the layout. Semantic information is useful to subdivide large text pages into smaller sections. Layout information is required to scale the components of a multi media document. To extract this information which is solely based on the document is very difficult. On one hand, HTML tags that could code the structure of a document are used for designing the text style, on the other hand, layout information can be distributed over many HTML pages (in case of frames) and is not available on the proxy side unless the proxy stores the access history and establishes a session concept.

4.2. Extensions of AdaWeb concept

In order to improve the mobile WWW access, we have extended our system concept with new components. These components are:

- Client Proxy including Client Monitor and Data Decompressor.
- Data Compressor.
- Resource Trader.

4.2.1. Data Compression and client components

The Data Compression/Decompression is related to the integration of a client-side component due to the usage of a proprietary coding algorithm which is not supported by standard browsers. Our coding algorithm is based on the GZIP coding scheme [18,19], but, instead of generating the whole dictionary, an initial dictionary depending on the transmitted content type is predefined [20]. This dictionary is known by the data compressor as well as the decompressor. The advantage is that the dictionary does not have to be transferred, thus avoiding the ineffective phase at the beginning of the data compression. In this way, relatively small data files can be effectively compressed.

The compression component of AdaWeb starts after the adaptation process is finished. Like the other components of the AdaWeb proxy, it communicates with the RIB to check which decompression possibilities the mobile client disposes of. If an adequate dictionary for the current content type is supported on the client side, the compression starts. Otherwise, the data stream is sent to the client without modification.

A compression algorithm as mentioned above necessitates decompression as a counterpart. The decompression is one component running on the mobile client

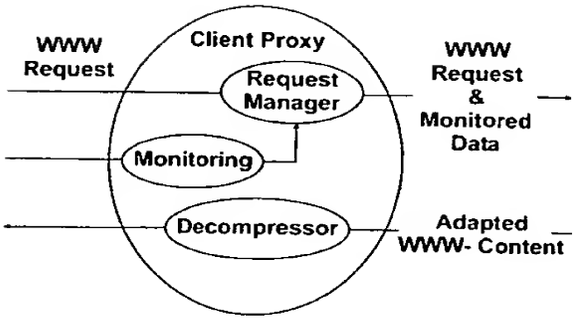

Fig. 5. Components at the client side.

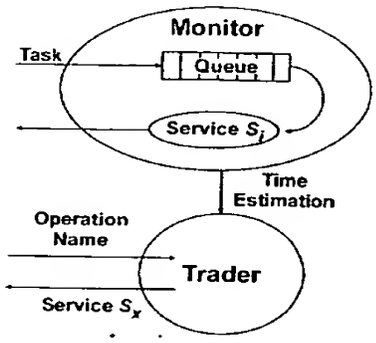

Fig. 6. Interplay of service monitor and trader.

and is integrated within a client-side proxy. On one side, this proxy transmits requests from the browser to AdaWeb. On the other side, it receives the adapted WWW content from AdaWeb and when the content is compressed it is decompressed before being forwarded to the browser.

However, we do not only use the client-side proxy for modification of the data transfer from AdaWeb to the proxy, but also for getting additional dynamic context information from the client. This information includes, for example, the battery life of the mobile device, the workload of its memory/processor or the current position of the user. A Monitoring Component is responsible for collecting this information and sending it to the Request Manager which appends it in the form of a cookie on a transmitted request to AdaWeb (see Fig. 5).

4.2.2. Trader component

When reaching a valid premise of a rule the Adaptation Component of AdaWeb executes the operations within the action part of the rule. The execution of an operation in this case means that the Trader Component is at first invoked with the name of the operation. Returned to the Adaptation Component, the result of this invocation is a service that can, at present, best perform the operation. The best service in our concept is the one which performs the desired operation in the shortest time.

Therefore, the Trader Component needs to know how long a service would require for the execution of the operation. In order to achieve this, we will encapsulate each service with a Monitor Component. This component receives requests like that of the service itself, but instead of executing a task directly it is buffered into a FIFO-queue and is delivered to the service when it reaches the beginning of the queue. For each task the Monitor measures the execution time and, taking these measurements into account, as well as the number of tasks inside the queue, it estimates the time in which the next received task will be finished.

At first, we will start our development with a simple averaging as estimation and then compare the results with a more sophisticated model, where the estimation is based on a Kalman filter and considers dynamic variations.

The result, the estimation of the execution time for the next task, is transferred from the Monitor to the Trader. The Trader manages similar services within a priority list ordered by the time value. When requesting a service by the Adaptation Component, the first service within the list is returned. Fig. 6 shows the interplay of the components.

5. Integration of AdaWeb within mobile applications

AdaWeb can manage a lot of dynamic context information and is therefore suitable for context-aware WWW applications. This enables information access and information presentation taking into consideration the current situation of the user. The goal is to make the interaction with the WWW as simple as possible, so that the user can concentrate on the current task instead of the handling of the device. In this regard context information helps to decide which and how the information will be presented.

The location of a user is an important context parameter, especially for mobile application scenarios. For this reason the additional proxy component on the client side is connected to a GPS-receiver for detecting the current position.

With this extension AdaWeb is one of the main components of our city information system, as the architecture shows (see Fig. 7). This system informs the user about his current position and allows him to make requests related to his environment; for example where the nearest restaurant is.

AdaWeb is responsible for managing the context information and adapting the WWW-based user interface of the system to the display capabilities of the mobile device. To fulfill the requirements it monitors the current

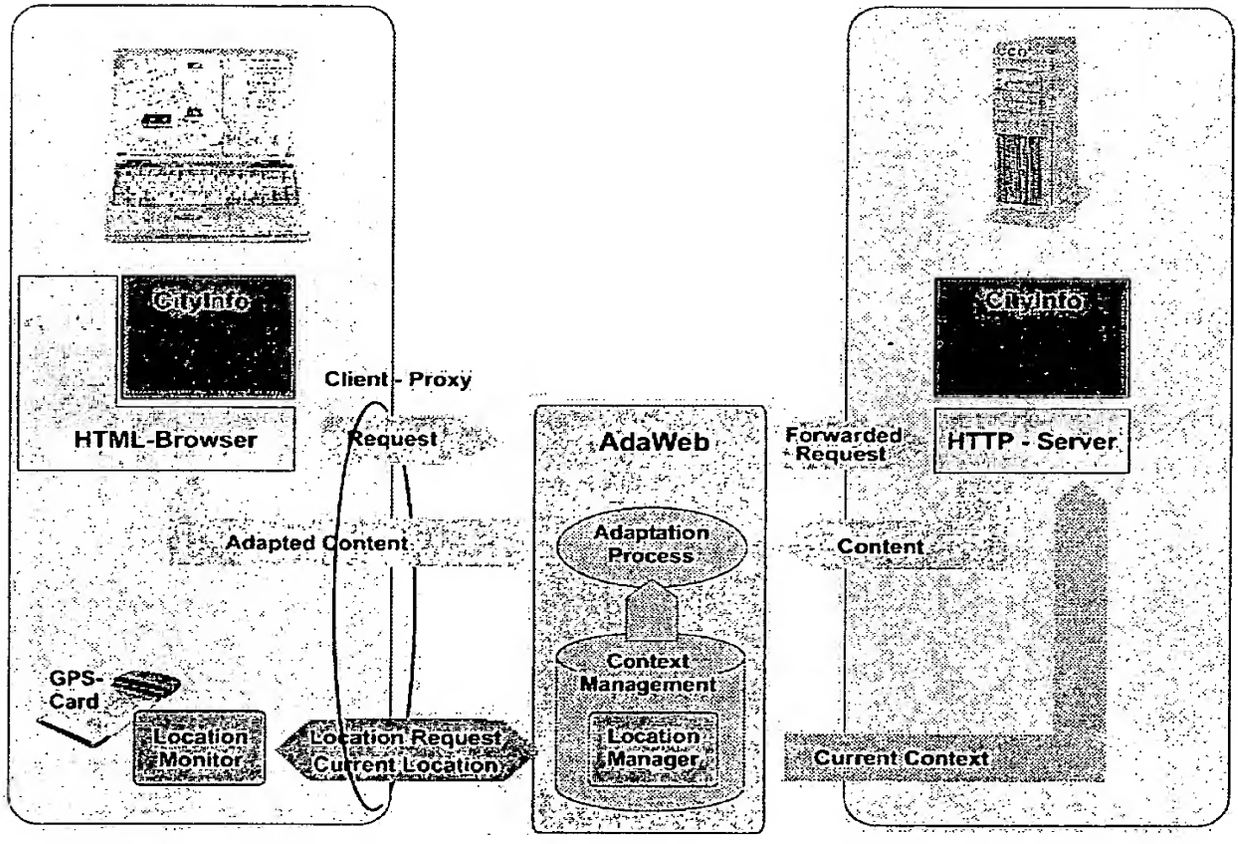

Fig. 7. Architecture of the WWW-based City Information System.

position of the user. With the context information from AdaWeb, the city information server generates the map where the current position of the user is marked with a circle. Furthermore, the map contains highlighted objects that are interesting for the user. Depending on the capabilities of the device, the objects are displayed in 3D presentations or in the form of rectangles, which link to further information.

In a further development phase, the system will send messages autonomously, if it deduces from an entry within the user profile that the user could be interested in it. If the user profile contains for example the key "interested in" which may refer to "sight seeing", a short text informs the user of any tourist attraction in the vicinity. When the user has arrived at his chosen destination, he is able to enjoy his sightseeing, as the information he needs is given in an acoustical form.

6. Conclusion

The vision of a simple and ubiquitous information access requires solutions for an efficient integration of mobile devices within the WWW infrastructure. The main problems are the restricted resources on the mobile computing device and the narrow bandwidth and unreliability of the wireless link.

In this paper we have presented a system concept enabling a context-aware adaptation for the information access and presentation within mobile WWW applications. A rule-based system is used to evaluate dynamically the most appropriate adaptation strategy for the current context. The context is defined by user preferences and device properties. The basis for the system is the Resource Information Base (RIB) that maintains the information about the device properties, user preferences and adaptation rules.

The system has been developed as a single-proxy architecture that performs the adaptation of the WWW access with the resources of hosts in the fixed network. This approach has the advantage that no changes have to be made on the browser at the client side. This means that every mobile computing device can use its standard browser to gain access to arbitrary WWW-servers via AdaWeb.

First experiences with the core system show that AdaWeb is a very promising solution and a first step in

the right direction to adapt arbitrary WWW content according to the properties of the end device and user preferences. In order to consider the aspects of data traffic, network latency and unreliability additional components on the client as well as on the proxy are needed. We have described in this paper how a client component can easily be integrated within the AdaWeb system to extend the managed context information. The integration of a compression and a specific location component are especially illustrated. Here, a locator for the Global Positioning System is used to monitor the location of a mobile user. It enables AdaWeb to utilise location information for the adaptation process within location-dependent applications. Moreover, we have presented a trader concept for AdaWeb that identifies the available resources for the adaptation process. The selection of free resources for the adaptation pipeline supports a scalable adaptation service by distributing the adaptation workload.

Obviously, several interesting issues still remain open. Currently, implementations of several components, such as filters and compression concepts for further media representations and the trader concept are in progress to validate the AdaWeb framework.

Besides the dynamic adaptation, one further strength of AdaWeb is its ability for customisation, because the mobile user can select the adaptation strategy with its preferences. The main goal is to enable an intuitive control of the context adaptation process for the user via a user interface. This could be achieved through creating sets of rules, which are based on pre-defined mobile situations (e.g., the type of motion — on foot, by car, etc.) and pre-selected user options.

We are confident that besides mobile devices other information appliances will also benefit from AdaWeb. One example is the presentation of Web content via AdaWeb on a TV set.

References

[1] Fox A, Brewer EA. Reducing WWW latency and bandwidth requirements by real-time distillation. Fifth International Conference on the World Wide Web, Computer Networks and ISDN Systems, vol. 28(7–1), 1996. p. 1445.

[2] Shimada T, et al. Interactive scaling control mechanism for world wide web systems. Sixth International Conference on the World Wide Web, Santa Clara, 1997. p. 491.

[3] Lauff M, Gellersen HW. Multimedia Client Implementation on Personal Digital Assistant, IDMS '97, Fourth International Workshop, Darmstadt, Germany, 1997.

[4] Kamada T. Compact HTML for Small Information Appliances, W3C note, http://www.w3.org/TR/1998/NOTE-compactHTML-19980209, 1998.

[5] Scott Palmer. WAP Forum, http://www.wapforum.org, 1998.

[6] Fox A, Goldberg I, Gribble SD, Lee DC, Polito A, Brewer EA. Experience With Top Gun Wingman. In: Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98), Lake District, UK, 1998.

[7] Mazer DM. Disconnected and weakly connected access to the World Wide Web: issues and techniques. In: Tutorial 5, The Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 97), Budapest, 1997.

[8] Schilit BN, Douglis F, Kristol DM, Krzyzanowski P, Sienicki J, Trotter JA. TeleWeb: loosely connected access to the World Wide Web. Computer Networks and ISDN Systems 1997;28(7–11):1431.

[9] Liljeberg M, Alanko T, Kojo M, Laamanen H, Raatikainen K. Optimizing World-Wide Web for weakly connected mobile workstations: an indirect approach. In: Second International Workshop on Services in Distributed and Networked Environments (SDNE '95), Whistler, Canada, 1995.

[10] Baquero C, Fonte V, Oliveira R. WWW browsing under disconnected and semi-connected operation. In: Portuguese WWW National Conference, Braga, 1995.

[11] Watson T. Application design for wireless computing. In: Workshop on Mobile Computing Systems and Applications, Santa Cruz, December 1994. http://snapple.cs.washington.edu:600/wit/presentations.html.

[12] Fox A, Gribble SD, Chawathe Y, Brewer EA. Adapting to network and client variation using active proxies: lessons and perspectives. IEEE Personal Communciations (1998), to appear.

[13] Bickmore TW, Schilit BN. Digestor: device-independent access to the World Wide Web. In: Proceedings for the Sixth International World Wide Web Conference, 1997.

[14] Freytag C, Kumpf C, Neumann L. Utilisation of device and user profiles for mobile WWW-Access. Fifth International Workshop on Mobile Multimedia Communications (MoMuC '98), Berlin, 1998.

[15] Rauschenbach U. Progressive image transmission using levels of detail and region of interest. In: Proceedings of IASTED Conference on Computer Graphics and Imaging — CGIM '98, Halifax, Canada, 1998.

[16] Böningk J, von Lukas U. Interactive exchange of structured multimedia data with mobile hosts. Global Communication Interactive 97 (1997).

[17] The Gimp Homepage. http://www.gimp.org/, 1999.

[18] Storer JA. Data compression: methods and theory. Rockville, MD: Computer Science Press, 1988.

[19] Network Working Group; RFC 1950. ZLIB Compressed Data Format Specification. Version 3.3, 1996.

[20] Kumpf C. Kompression von Volumendaten. Studienarbeit, TH-Darmstadt. Darmstadt, 1995.